TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta mechatroniky a mezioborových inženýrských studií

DIPLOMOVÁ PRÁCE

ŘADIČ PRO PŘENOS DAT INFRAČERVENÝM ROZHRANÍM

Liberec 2002 Jan Halama

TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta mechatroniky a mezioborových inženýrských studií

Studijní program 2612T – Elektrotechnika a informatika

Studijní obor: Automatické řízení a inženýrská informatika

Katedra elektroniky a zpracování signálů

ŘADIČ PRO PŘENOS DAT INFRAČERVENÝM ROZHRANÍM

CONTROLLER FOR INFRARED DATA TRANSMISSION

Jan Halama

Vedoucí diplomové práce: Ing. Milan Kolář, Csc.

Konzultant diplomové práce: Ing. Zdeněk Plíva

ANOTACE

TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta mechatroniky a mezioborových inženýrských studií
Katedra elektroniky a zpracování signálů

Studijní program: 2612T – Elektrotechnika a informatika
Diplomant: Jan Halama
Téma práce: Řadič pro přenos dat infračerveným rozhraním
Theme of work: Controller for infrared data transmission
Rok obhajoby DP: 2002
Vedoucí DP:Ing. Milan Kolář, Csc.
Konzultant: Ing. Zdeněk Plíva

Resumé:

Cílem diplomové práce je vytvořit elektronický systém schopný přenášet data infračerveným rozhraním. Komunikační systém byl vytvořen na bázi programovatelných zakázkových obvodů a naprogramován jazykem VHDL vhodným pro návrhy složitých obvodů. Zdrojem informací při vytváření zařízení, které komunikuje po infračerveném rozhraní, byla norma IrDA. Vytvořený elektronický obvod obsahuje fyzickou a spojovou vrstvu komunikačního protokolu. Navržený komunikační obvod lze díky použité technologii a strukturovanému návrhu jednoduše začlenit do větších projektů využívajících infračervený přenos dat.

Abstract:

The aim of the diploma thesis is to create an electronic system, which will be able to transmit data by infrared interface. The communication system was created on the basis of the field programmable gate array and this system was pre-set by the VHDL language. This language is suitable for design difficult circuits. The source of information about infrared interface was IrDA standard. The created electronic circuit contains physical and data link layers of communication protocol. Due to the technology which was used and its structural design the communication circuit can be used in wide applications.

Poděkování

Na tomto místě bych chtěl poděkovat Ing. Milanu Kolářovi, Csc. za odborné vedení, pomoc při zpracování diplomové práce, cenné rady a poskytnuté informace.

Prohlášení

Byl jsem seznámen s tím, že na mou diplomovou práci se plně vztahuje zákon č.121/2000 o právu autorském, zejména § 60 (školní dílo) a § 35 (o nevýdělečném užití díla k vnitřní potřebě školy).

Beru na vědomí, že TUL má právo na uzavření licenční smlouvy o užití mé práce a prohlašuji, že **s o u h l a s í m** s případným užitím mé práce (prodej, zapůjčení apod.).

Jsem si vědom toho, že užít své diplomové práce či poskytnout licenci k jejímu využití mohu jen se souhlasem TUL, která má právo ode mne požadovat přiměřený příspěvek na úhradu nákladů, vynaložených univerzitou na vytvoření díla (až do jejich skutečné výše).

V Liberci 24. 5. 2002 …….

Jan Halama

Místopřísežné prohlášení

„Místopřísežně prohlašuji, že jsem diplomovou práci vypracoval samostatně s použitím uvedené literatury.“

V Liberci 24. 5. 2002 …….

Jan Halama

OBSAH

1. Úvod

Infračervený přenos dat je v současné době velmi rozšířený způsob bezdrátové komunikace. Infračervené rozhraní je využíváno v počítačové technice (tiskárny, skenery, digitální fotoaparáty), používá se také v mobilních telefonech. Velikou zásluhu na širokém rozšíření infračerveného rozhraní má společnost IrDA, která vydala soubor norem standardizující infračervený přenos dat.

Základní vlastností infračerveného rozhraní je bezdrátové spojení vysílače s přijímačem. Důvodem, proč vznikl požadavek na vytvoření řadiče pro přenos dat po infračerveném rozhraní, je projekt sběru dat z čidla, které může být umístěno na nedostupném místě. Aplikovat bezdrátový přenos dat je ideálním řešením tohoto problému. Řadič pro přenos dat po infračerveném rozhraní je tedy součástí většího projektu, který bude využívat služeb řadiče.

Úkolem diplomové práce je vytvořit elektronické zařízení zabezpečující komunikaci po infračerveném rozhraní, které bude vycházet z normy IrDA a co nejvíce se normě IrDA přiblíží. Dalším požadavkem je využít při popisu systému normy IrDA pro fyzickou vrstvu, která umožňuje komunikaci rychlostí 4 Mb/s.

Realizace řadiče pro přenos dat po infračerveném rozhraní má být provedena na bázi programovatelných zakázkových obvodů FPGA (*Field Programmable Gate Array*). Výhodou moderních FPGA obvodů je možnost zadávat funkce prostředky CAD. Obvody FPGA jsou vhodné i pro velmi složité funkce a umožňují vytvářet univerzální logické bloky, které lze dále spojovat. Této vlastnosti lze s výhodou využít při implementaci normy IrDA. Norma IrDA vychází ze sedmivrstvého modelu ISO/OSI a dělí přenos dat do několika samostatných vrstev. Lze tedy velice jednoduše při aplikaci infračerveného rozhraní v jiném projektu nahradit nejvyšší aplikační vrstvu řadiče přenosu dat a využít všechny ostatní vrstvy, jejichž funkce se nezmění. Obvod FPGA bude naprogramován v prostředí MAX+plus II jazykem VHDL, který je určen pro návrhy složitých zakázkových obvodů.

Ověření funkčnosti obvodů bude provedeno jednak v simulacích v prostředí MAX+plus II a dále na testovacím přípravku s programovatelným hradlovým polem.

2. Referenční model ISO/OSI

Referenční model ISO/OSI (jak uvádí Hlava [2, s.114] a Peterka [6]) je sedmivrstvým modelem komunikačního standardu. Model ISO/OSI tedy vychází z rozdělení komunikačního protokolu do sedmi vrstev. Každá vrstva je schopna komunikovat pouze s vrstvou jí podřízenou a nadřízenou. Struktura tohoto modelu je hierarchická – vyšší vrstva využívá funkcí poskytovaných vrstvou nižší a tím ji řídí. Z modelu ISO/OSI vycházejí komunikační standardy (např. IrDA), které pro různé typy komunikačních rozhraní definují konkrétní podobu jednotlivých vrstev.

2.1 Vrstvy modelu ISO/OSI

Sedm komunikačních vrstev referenčního modelu ISO/OSI je v této kapitole uspořádáno vzestupně (nejnižší je Fyzická vrstva a nejvyšší Aplikační vrstva).

Fyzická vrstva (*Physical Layer*)

Fyzická vrstva, definuje přenos jednotlivých bitů mezi příjemcem a odesílatelem prostřednictvím fyzické přenosové cesty, kterou tato vrstva přímo ovládá. Fyzická vrstva definuje standardy, které určují elektrické, mechanické, funkční a procedurální vlastnosti rozhraní, jako jsou napěťové úrovně, délka světelných impulsů, atd.

Norma IrDA definuje již ve fyzické vrstvě kódování dat 4PPM modulací, zabývá se uspořádáním dat v blocích (paketech) a definuje start a stop bity, čímž zasahuje do spojové vrstvy sedmivrstvého modelu ISO/OSI.

Spojová vrstva (*Data Link Layer*)

Spojová vrstva se zabývá bezchybným přenosem bloků dat, které označuje jako rámce (*frames*). Tato vrstva tedy musí definovat způsob nalezení začátku a konce rámce, čímž se zajišťuje správná interpretace řídicích a datových bitů. Dále zajišťuje bezchybný přenos dat. Rámec musí obsahovat kontrolní nebo samoopravnou posloupnost bitů, která umožní detekci a nebo i opravu případných chyb v datech (realizace např. pomocí cyklických kódů). Potřeba opravných nebo pouze detekčních posloupností bitů je dána povahou přenosu, která může být simplexní (pouze jeden směr přenosu dat), poloduplexní (obousměrný, ale v určitém čase možný přenos pouze jedním směrem) a duplexní (umožňuje komunikovat oběma směry

najednou). V poloduplexním a duplexním režimu je zde možnost při detekci chyby poslat zpět požadavek na opětovné vyslání celého rámce znovu.

Dále tato vrstva provádí řízení toku dat a navazuje komunikaci mezi přijímačem a vysílačem. Vysílající musí respektovat připravenost přijímače k přijímání dat, aby ho nezahltil. Na úrovni linkové vrstvy je to řešeno pozastavením vysílání dalšího rámce.

Existují dva přístupy k přenosu dat – synchronní a asynchronní. Nevýhodou asynchronního přístupu je potřeba start a stop bitů, čímž se snižuje celková přenosová rychlost.

Síťová vrstva (*Network Layer*)

Protože linková vrstva je schopna zabezpečit přenos rámců pouze mezi uzly, mezi kterými je přímé spojení, je zapotřebí síťové vrstvy, která zabezpečí směrování rámců, které se v síťové vrstvě označují jako pakety (*pakets*).

Transportní vrstva (*Transport Layer*)

Transportní vrstva má za úkol rozdělovat data do paketů při režimu vysílání a v režimu přijímání skládá pakety ve správném pořadí dohromady.

Relační vrstva (*Session Layer*)

Tato vrstva má za úkol navazovat spojení mezi 2 koncovými účastníky. Dále pak například v poloduplexním režimu koordinuje vysílání a přijímání obou dvou účastníků komunikace.

Prezentační vrstva (*Presentation Layer*)

Prezentační vrstva provádí konverze přenášených dat, pokud mezi různými systémy existují rozdíly v kódování dat (např. rozdílné kódování znaků ASCII/EBCDIC).

Aplikační vrstva (*Application Layer*)

Definuje obecně použitelné mechanismy společné pro více aplikací. Např. realizace vlastního předávání zpráv v síti pomocí elektronické pošty. Tato vrstva se již ale nezabývá prostředím, ve kterém toto předávání bude probíhat.

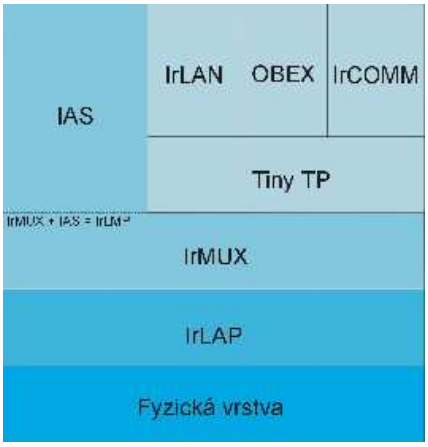
3. Základní informace o IrDA (Infrared Data Association)

Společnost IrDA byla vytvořena jako nevýdělečná organizace v roce 1993 z podmětu 50 celosvětových firem. Sídlo IrDA je ve Walnut Creek v Kalifornii. V současné době již IrDA sdružuje více než 150 průmyslově založených společností. Normy IrDA podporuje široké množství přístrojů, počítačových a komunikačních zařízení.

Úkolem IrDA je vytvářet a propagovat normy pro levný infračervený přenos dat, který vychází ze sedmivrstvého modelu přenosu dat ISO/OSI.

Již v roce svého založení IrDA vydala první základní normu označenou SIR (Serial Infrared), která definuje vlastnosti fyzické vrstvy infračerveného přenosu dat. V roce 1994 IrDA publikovala balík norem, který obsahuje normy SIR, IrLAP (Serial Infrared Link Access Protocol), IrLMP (Link Management Protocol). V roce 1995 byla norma SIR rozšířena o část definující přenos rychlostí 4 Mb/s a ve stejném roce společnost Microsoft oznámila, že operační systém Windows 95 podporuje spojení se zařízeními komunikujícími po IrDA rozhraní. V dalších letech IrDA dále pracovala na vylepšení přenosu dat po infračerveném rozhraní. Dokladem toho je norma pro přenos dat rychlostí 16 Mb/s, dále norma IrOBEX (Object Exchange Protocol) pro speciální API vrstvu, která je použitelná jak pro IrDA komunikační zařízení, tak pro zařízení komunikující po konkurenčním bezdrátovém rozhraní Bluetooth.

Komunikační protokoly vydané společností IrDA jsou sendvičového typu. Jednotlivé protokoly jsou sobě nadřazeny. Jsou to samostatné bloky, které spolu komunikují pomocí svých vstupů a výstupů. Tato struktura vychází ze sedmivrstvého modelu ISO/OSI. Přesná struktura normy IrDA je vidět na obr. 1.



Obr. 1 – Struktura normy IrDA

Fyzická vrstva, vrstvy IrLAP a IrLMP jsou povinnými vrstvami, které musí být implementovány, pokud má být dosáhnuto nejzákladnějšího přenosu dat s jiným komunikačním zařízením komunikujícím po infračerveném rozhraní IrDA. Tyto vrstvy jsou podrobněji popsány v dalších kapitolách.

Vrstva Tiny TP slouží ke kontrole toku dat za vrstvou IrLMP. Vrstva IrLMP obsahuje také mechanizmy pro řízení toku dat, ale pokud je v jednu chvíli řízeno více spojení a jedna z vrstev IrLAP požádá o spojení, pak je druhé spojení IrLAP úplně odříznuto. Tento problém řeší vrstva Tiny TP.

Vrstva IrOBEX slouží k zjednodušení výměny dat mezi zařízeními různých typů. Data jsou ve vrstvě IrOBEX reprezentována objekty. Tyto objekty mohou obsahovat data různých typů od běžných souborů, digitálních obrázků přes databázová data až po telefonní zprávy.

Vrstva IrCOMM (Serial and Parallel Port Emulation) slouží k simulaci chování sériového rozhraní RS232 a paralelního rozhraní. Tato vrstva je vytvořena proto, aby existující PC aplikace byly schopny komunikovat po infračerveném rozhraní, aniž by byly změněny. Novější aplikace již využívají přímo vyšších vrstev IrDA, protože vrstva IrCOMM zamaskuje mnoho dobrých vlastností rozhraní IrDA.

Vrstva IrLAN umožňuje připojit počítač k síti pomocí IR LAN adaptéru. IR LAN adaptér je hardwarové zařízení vyvinuté například společností Hewlett Packard. Dále dovoluje komunikovat dvěma počítačům mezi sebou přes další komunikační zařízení v síti. Počítač lze také pomocí vrstvy IrLAN připojit k síti prostřednictvím již připojeného zařízení.

Další informace uvádí Magowan [5] a Angerstein [1].

4. Fyzická vrstva normy IrDA

Bez speciálního komunikačního protokolu by nebyla komunikace po infračerveném rozhraní dostatečně robustní. Norma IrDA popisuje směrnice pro spojení zařízení pomocí infračerveného rozhraní.

Komunikace po infračerveném rozhraní je z principu poloduplexní. Zařízení není schopno zároveň přijímat a vysílat, protože není opticky izolováno. V případě plného duplexu by vysílaný signál interferoval se signálem přijímaným. Přenos po infračerveném rozhraní je také limitován časovou prodlevou mezi koncem vysílání a dobou kdy je zařízení schopno začít přijímat. Tato doba je dána saturací přijímače v době kdy zařízení vysílá. Norma IrDA tuto dobu od konce vysílání do doby, kdy přijímač získá svou plnou citlivost, stanovuje na max. 10 ms.

Pro splnění normy IrDA musí zařízení zabezpečovat bezchybnou komunikaci ve vzdálenosti (vysílač - přijímač) 0 až 1 m při úhlu vysílání 0 až ±15° od osy vysílače.

Součástí normy jsou předpisy pro modulaci, intenzitu záření, optickou citlivost přijímače, přenosovou rychlost a šumovou imunitu.

Přenosové rychlosti, které popisuje norma IrDA jsou v rozsahu 2,4 Kb/s – 4 Mb/s.

4.1 Popis jednotlivých částí rámce pro přenosovou rychlost 4 Mb/s

Při rychlosti 4 Mb/s se využívá asynchronního přenosu dat po rámcích (paketech). Pro přenos rychlostí 4 Mb/s se používá paket v následujícím uspořádání:

PA	STA	DD	STO

Vysílání probíhá zleva, tzn., že nejprve se vysílá část PA, nakonec část STO.

PA (*Preamble Field*)

PA je úvodní sekvence bitů sloužící k synchronizaci hodinového signálu přijímače s hodinovým signálem vysílače. Během přijímání PA přijímač čeká na následující posloupnost STA. Sekvence PA se skládá z 16x se opakující posloupnosti šestnácti bitů:

1000 0000 1010 1000

Vysílání probíhá zleva (1. vysílaný bit) doprava (poslední vysílaný bit).

STA (*Start Flag*)

STA je část paketu, která slouží k detekci začátku paketu. Následující část DD již obsahuje vlastní přenášená data. STA je tvořena touto konstantní posloupností 32 bitů:

0000 1100 0000 1100 0110 0000 0110 0000

STO (*Stop Flag*)

STO je sekvence bitů, která ukončuje daný paket. Následuje okamžitě po části DD a je tvořena následující posloupností 32 bitů:

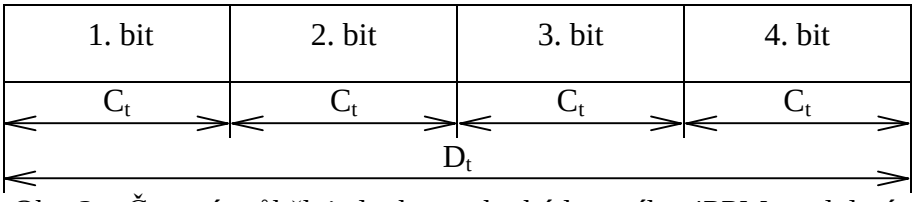
0000 1100 0000 1100 0000 0110 0000 0110

DD (*Data Symbol*)

DD je hlavní částí paketu, která obsahuje užitečná data. Počet bajtů není přesně definovaný, lze je tedy nastavit podle aktuální potřeby. Pouze platí omezení na maximální počet 2048 bajtů v jednom poli DD. Při větším počtu bajtů se zvyšuje teoretická přenosová rychlost, protože klesá počet vysílaných paketů a tudíž i částí PA, STA a STO (které nenesou žádnou užitečnou informaci). Dojde-li ale při přenosu ke vzniku chyby, je nutno vyslat opětovně celý paket. Při velké délce paketu to může výrazně snížit reálnou rychlost celého přenosu. Je tedy vhodné nastavit počet bajtů vysílaných v jednom paketu s ohledem na chybovost přenosu a množství přenášených dat.

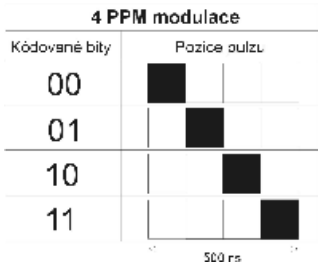
Veškerý obsah části DD je kódován bezpečnostním kódem. Při rychlosti 4 Mb/s se používá kód 4PPM (*Four Pulse Position Modulation*). Princip spočívá v tom, že vstupní data jsou rozdělena do po sobě jdoucích dvojic a podle binární hodnoty této dvojice se mění poloha log. jedničky odpovídající výstupní čtveřice od 1. až do 4. bitu. Kód tedy zdvojnásobuje množství dat v poli DD oproti vstupní informaci, jenž hodláme přenést. Výhoda je v tom, že umožňuje detekovat chybu vzniklou při přenosu (s vyjímkou nepravděpodobné chyby posunutí pulsu log. jedničky v čase). Další výhoda 4PPM modulace je v tom, že se přijímači lépe udržuje úroveň okolního osvětlení, protože na něj dopadá konstantní počet optických pulsů za danou dobu.

Na obr. 2 je časové zobrazení jednoho znaku (2 bity) zakódovaného 4PPM modulací do 4 bitů. Tento jeden znak se při přenosové rychlosti 4Mb/s vysílá po dobu D_t = 500 ns. Doba pro jeden výstupní bit je definována jako $C_t = D_t / 4$ = 125 ns. Na obr. 3 je způsob kódování dvojice bitů do 4PPM kódu.



Obr. 2 – Časový průběh jednoho znaku kódovaného 4PPM modulací.

Vstupní data	4PPM kód
00	1000
01	0100
10	0010
11	0001



Obr. 3 - Způsob kódování dvojice vstupních bitů 4PPM modulací.

Z obr. 3 je zřejmé, že log. jednička je na výstupu vždy právě v jeden výstupní bit ze čtveřice. Jednotlivé dvojice vstupních dat jsou navíc při kódování bajtu přehozeny, postup demonstruje obr. 4 pro příklad vstupního slova ‘1B’ v hexadecimálním kódu (toto číslo obsahuje všechny možné kombinace dvojic bitů). Dále jsou pro příklad uvedeny zakódované bajty 0Bh a A4h. Vysílání opět probíhá zleva (první vysílaný bit) doprava (poslední vysílaný bit).

Vstupní bajt v hexadecimálním kódu	vstupní bajt binárně	Zakódovaný vstupní bajt v kódu 4PPM
1Bh	00 01 10 11	
		0001
		0010
		0100
		1000
		0001 0010 0100 1000
0Bh	00 00 10 11	0001 0010 1000 1000
A4h	10 10 01 00	1000 0100 0010 0010

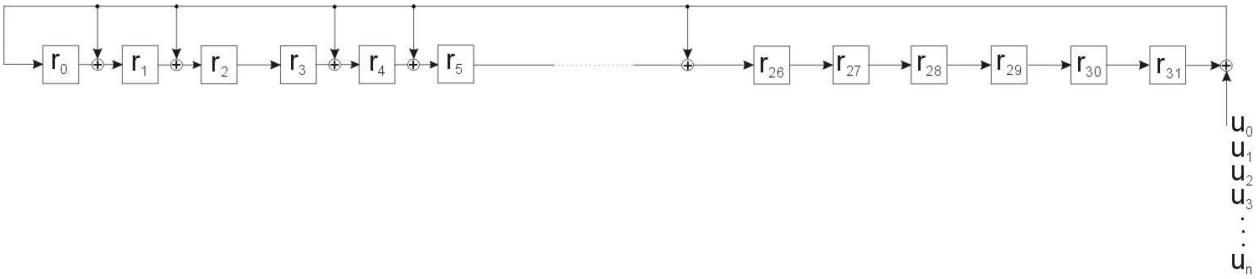
Obr. 4 – Princip kódování 4PPM modulace na konkrétních příkladech.

Součástí pole DD je kromě platných dat i nástroj pro kontrolu přenesených dat. Jedná se o 32-bitový CRC (*cyclic redundancy check*) kód. CRC kód se generuje před 4PPM modulací a je pomocí 4PPM modulace tak jako platná data zakódován. Čtyři bajty CRC kódu se připojují na konec bloku DD za platná data.

Algoritmus výpočtu 32 bitového CRC kódu popisuje norma IEEE 802 CRC32. CRC kódy popsal Hlavička [3]. Polynom CRC32 je definován takto:

$$CRC(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x + 1$$

Před vlastním výpočtem je celý registr CRC přednastaven na samé logické jedničky. CRC registr je posuvný registr se zavedenými zpětnými vazbami (viz obr. č. 5) podle CRC32 polynomu. Po vstupu všech datových bitů se na vstup registru přivede logická nula a na výstup posuvného registru se postupně dostane 32 bitů CRC kódu. CRC generátorem se realizuje dělení polynomu vstupních dat generujícím polynomem, kontrolním CRC součtem se pak stává zbytek po tomto dělení.



Obr. 5 – 32-bitový CRC generátor

Přesný popis fyzické vrstvy vytvořil Petrolka [7].

5. Spojová vrstva normy IrDA

Norma IrDA spojovou vrstvu dělí do dvou částí a řeší pouze řízení toku dat a navázání komunikace. Definice struktury paketu a zabezpečení dat kódováním je součástí fyzické vrstvy.

Navazování komunikace je popsáno v normě IrDA v části nazvané Protokol pro navázání spojení po sériovém infračerveném rozhraní (*Serial Infrared Link Access Protocol*). Řízení toku dat řeší Protokol pro řízení spojové vrstvy (*Link Management Protocol*).

5.1 Protokol pro navázání spojení po sériovém infračerveném rozhraní (IrLAP)

Tato část normy IrDA popisuje funkce, vlastnosti, protokol a služby pro spojení mezi počítači na úrovni spojové vrstvy. Vychází z již existujících poloduplexních asynchronních protokolů HDLC a SDLC.

5.1.1 Služby poskytované IrLAP vyšším vrstvám

Služby IrLAP, které slouží ke komunikaci s vyššími vrstvami se dělí na čtyři typy (obr. 6) podle směru toku dat a stavu IrLAP. IrLAP může být ve stavu, kdy přijímá a nebo vysílá rámec dat.

Požadavek (*Request*)

Vyšší vrstva vysílá požadavek pro vyvolání služby IrLAP.

Oznámení (*Indication*)

IrLAP vysílá oznámení vyšší vrstvě a tím jí informuje o události, nebo o zahájení akce.

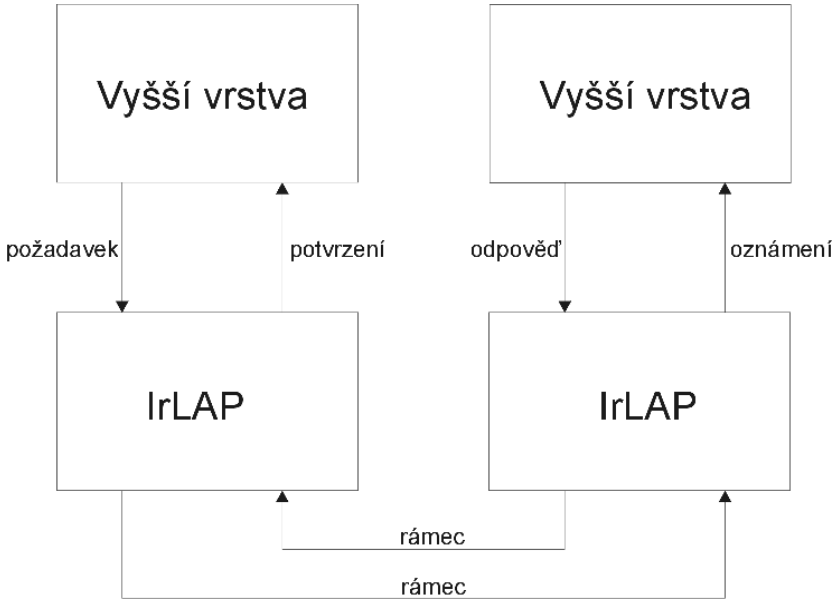
Odpověď (*Response*)

Vyšší vrstva vysílá odpověď na oznámení a tím potvrzuje, že oznámení bere na vědomí.

Potvrzení (*Confirm*)

IrLAP pomocí potvrzení informuje vyšší vrstvu o výsledku předchozího požadavku o službu.

Služby IrLAP vyšším vrstvám se dále dělí na služby poskytované před navázáním spojení a po navázání spojení.



Obr. 6 – Typy služeb IrLAP

Služby poskytované před navázáním spojení

1. Služby vyhledání (Discovery Services)

   IrLAP_Discovery.Request

   Vyšší vrstva požaduje od IrLAP zjistit zda je v dosahu jiné komunikační zařízení.

   IrLAP_Discovery.Confirm (*seznam komunikačních zařízení*)

   IrLAP vrací výsledek na požadavek vyhledání okolních komunikačních zařízení (IrLAP_Discovery.Request) v parametru *seznam komunikačních zařízení*.

   IrLAP_Discovery.Indication (*komunikační zařízení*)

   IrLAP informuje vyšší vrstvu o zařízení, které s IrLAP navazuje spojení v parametru *komunikační zařízení.*

   Parametr *komunikační zařízení* obsahuje tyto informace o komunikačním zařízení:

   Adresa zařízení - 32-bitová adresa zařízení

   Verze IrLAP - verze IrLAP - číslo v rozsahu 0 .. 255

   Informace - až 32 bajtů veliké pole definované vyšší vrstvou

2. Služby konfliktu adresy (Address Conflict Services)

Služby konfliktu adresy mají za úkol vyřešit problém kolize adres. Pokud v seznamu komunikačních zařízení budou existovat dvě zařízení se stejnými adresami, musí se zavolat služby konfliktu adres.

IrLAP_New_Address.Request (*adresa zařízení*)

Pokud dojde ke konfliktu adres vyžádá si komunikační zařízení řídící komunikaci jejich přegenerování.

IrLAP_New_Address.Confirm (*seznam komunikačních zařízení*)

Tato služba je výsledkem požadavku na přegenerování adres komunikačních zařízení. Vrací stejně jako služba IrLAP_Discovery.Confirm výsledek v parametru *seznam komunikačních zařízení.*

3. Služby přenosu dat bez spojení (Unit Data Services)

Přenos dat bez spojení je nespolehlivý. Probíhá bez určení adresy příjemce.

IrLAP_UnitData.request (d*ata*)

Požadavek vyšší vrstvy pro přenos dat.

IrLAP_UnitData.indication (d*ata*)

Pomocí této služby předává IrLAP data vyšší vrstvě.

Parametr *Data* obsahuje až 384 bajtů dat.

Služby poskytované po navázání spojení

1. Spojové služby (Connect Services)

IrLAP_Connect.Request (*Cílová adresa zařízení, Požadovaný-QOS*)

Tato spojová služba požaduje vytvoření spojení s komunikačním zařízením s adresou odpovídající parametru *Cílová adresa zařízení* a kvalitě přenosu odpovídající parametru *Požadovaný-QOS.*

IrLAP_Connect.Indication (*Zdrojová adresa zař., Ukazatel spojení, Přijatý-QOS*)

IrLAP pomocí této služby předává vyšší vrstvě adresu komunikačního zařízení, které požaduje spojení a kvalitu spojení v parametru *Přijatý-QOS*.

IrLAP_Connect.Response (*Zdrojová adresa zař., Ukazatel spojení, Požadovaný-QOS*)

Vyšší vrstva schválí touto službou spojení.

IrLAP_Connect.Confirm (*Ukazatel spojení, Přijatý-QOS*)

IrLAP informuje touto cestou vyšší vrstvu o správném navázání spojení s kvalitou spojení, která je uvedena v parametru *Přijatý-QOS*. Všechny ostatní služby se nyní budou odkazovat na toto spojení přes *Ukazatel spojení*.

Parametry *Přijatý-QOS* a *Požadovaný-QOS* obsahují informace o rychlosti spojení, prahu zrušení spojení, velikosti datové oblasti v rámci.

2. Služby přenosu dat (Data Services)

   IrLAP_Data.Request (*Ukazatel spojení*, *Data*, *Flag uspořádanosti*)

   Data mohou být přenášena podle parametru *Flag uspořádanosti* buď spořádaně a nebo neuspořádaně (rychleji – méně spolehlivý přenos). Parametr *Data* udává počet bajtů přenášených dat.

   IrLAP_Data.Indication (*Ukazatel spojení*, *Data*, *Flag uspořádanosti*)

3. Služby stavu (Status Services)

   Slouží vyšší vrstvě k přehledu o chybovosti přenosu.

   IrLAP_Status.request(*Ukazatel spojení*)

   IrLAP_Status.Indication(*Ukazatel spojení*, *Kvalita spojení*)

   IrLAP_Status.Confirm(*Ukazatel spojení, Flag chybně přijatého rámce*)

4. Inicializační služby (Reset Services)

   Inicializační služby způsobí, že se všechny zálohované nepřenesené rámce ztratí. Dojde k vynulování všech čítačů. K inicializaci dojde pouze v případě, že s ním souhlasí obě strany.

   IrLAP_Reset.Request (*Ukazatel spojení*)

   IrLAP_Reset.Indication (*Ukazatel spojení*)

   IrLAP_Reset.Response (*Ukazatel spojení*, *přijato*)

   IrLAP_Reset.Confirm (*Ukazatel spojení*, *přijato*)

   Parametr *přijato* informuje o tom, zda je akceptován požadavek na reset přenosu.

5. Služby zrušení spojení (Disconnection Services)

   IrLAP_Disconnect.request(*Ukazatel spojení*)

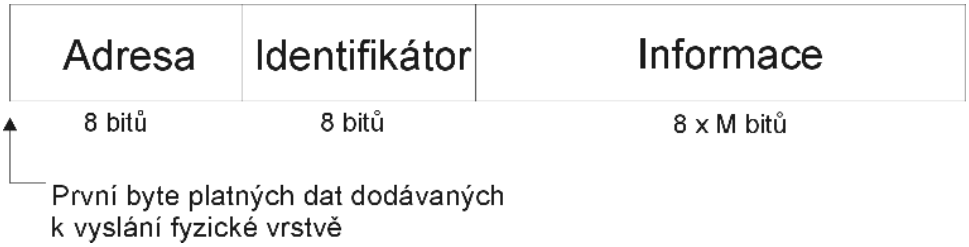
   Tato služba zruší spojení mezi dvěma komunikačními zařízeními a všechna neuložená data budou ztracena.

   IrLAP_Disconnect.indication(*Ukazatel spojení*)

Není třeba potvrzení zrušení spojení. To je vždy úspěšné.

5.1.2 Struktura rámce protokolu IrLAP

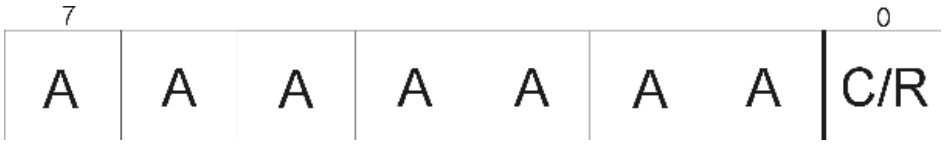
Každý rámec IrLAP obsahuje tři základní pole (obr. 7).



Obr. 7 – Struktura rámce IrLAP

Adresa

Pole adresa obsahuje identifikaci sekundární stanice (primární stanice navazuje komunikaci – je iniciátorem komunikace). Struktura tohoto pole je na obr. 8.



Obr. 8 – Adresové pole rámce IrLAP

Sedm bitů adresového rámce označené A obsahuje aktuální adresu sekundární stanice. C/R (command/response) bit je nastaven do logické jedničky pokud je odesílatelem rámce primární stanice. Potom je tento rámec označen jako řídící. Pokud je C/R bit nastaven do logické nuly, pak je vyslán sekundární stanicí a je označen jako rámec odpovědi.

Adresa B’1111111’ je rezervována jako globální adresa a používá se pro přenos dat bez navázání spojení (viz výše).

Identifikátor

Pole identifikátor obsahuje informace o funkci rámce. Rámce se podle funkce dělí na nečíslované rámce, kontrolní rámce a informační rámce.

1. Nečíslovaný rámec

   Tento typ rámce se používá k navázání a zrušení spojení, zprávám o chybách a přenosu dat (pokud nezáleží na pozici dat mezi ostatními rámci).
2. Kontrolní rámec

Kontrolní rámec pomáhá při přenášení informace, přestože sám žádné informační pole nemá. Používá se k odpovědím na správně nebo chybně přijaté rámce.

3. Informační rámec
   Informační rámce se používají k přenosu dat. Jednotlivé přenášené rámce se číslují a tak se zajišťuje správná reprezentace dat na straně přijímače.

Informace

Pole informace obsahuje vlastní data, která jsou rámcem přenášena. Kontrolní rámce toto pole vůbec neobsahují. Toto pole nemá definovanou délku, ale jeho velikost v bitech musí být násobkem osmi.

IrLAP popsal Williams [9].

5.2 Protokol pro řízení spojové vrstvy (IrLMP)

Součástí vrstvy IrLMP je vrstva IrMUX a IAS. IrMUX je multiplexor, který slouží k přepínání mezi jednotlivými spojeními (spojení na úrovni IrLAP). Dále má za úkol obstarávat vyšší úroveň vyhledávání okolních komunikačních zařízení. IrMUX informuje IrLAP, že je třeba znovu inicializovat adresy připojených zařízení, pokud došlo k jejich konfliktu.

Druhá část vrstvy IrLMP je vrstva IAS je také označována jako „žluté stránky“. Tato vrstva popisuje služby poskytované zařízením. IAS může být ve stavu klient, kdy odpovídá na otázky druhé strany a nebo ve stavu server, kdy se naopak ptá, jaké služby druhé zařízení poskytuje. K rámci vrstvy IrLAP přidává další bajt dat, který slouží pro přenos řídicích informací mezi vrstvami IAS.

IrLMP popsal Seaborne [8].

6. Návrh a realizace fyzické vrstvy

Vytvořený blok fyzické vrstvy plně odpovídá normě IrDA 4Mb/s. Vysílané a přijímané pakety dat splňují přesně strukturu definovanou normou IrDA.

Pro správné přijímání a vysílání paketů musí vrstva také zajišťovat řízení synchronizace hodin přijímače s vysílačem a řízení integrovaného obvodu IRMS6400 (viz IRMS6400 Datasheet [4]), který zajišťuje vysílání infračervených pulsů pomocí IR diod.

6.1 Vstupy a výstupy fyzické vrstvy

Vstupy a výstupy fyzické vrstvy se dělí do tří částí. Vstupy a výstupy (dále jen I/O) sloužící ke komunikaci s vyšší linkovou vrstvou, I/O řídicí synchronizaci hodin přijímače s vysílačem a I/O, které řídí integrovaný obvod IRMS6400.

I/O pro komunikaci s linkovou vrstvou

signál	vstup / výstup I / O	popis signálu
DATA_IN	I	vstupní 8 bitová datová sběrnice
DATA_OUT	O	výstupní 8 bitová datová sběrnice
CLK	I	hodinový signál
RESET	I	resetovací signál
PAKET_OK	O	správně přijatý paket
PAKET_ER	O	nesprávně přijatý paket
DATA_READ	O	čtení dat z DATA_IN
DATA_WRITE	O	platná data na DATA_OUT
BVPAKETU_SEL	I	počet datových bajtů v paketu
TLAC	I	start vyslání jednoho paketu

I/O řídicí synchronizaci hodin

signál	vstup / výstup I / O	popis signálu
PAG	O	generovaná synchronizační směs
PAR	O	přijímaná synchronizační směs
AP	O	řízení analogového přepínače

I/O řídicí integrovaný obvod IRMS6400

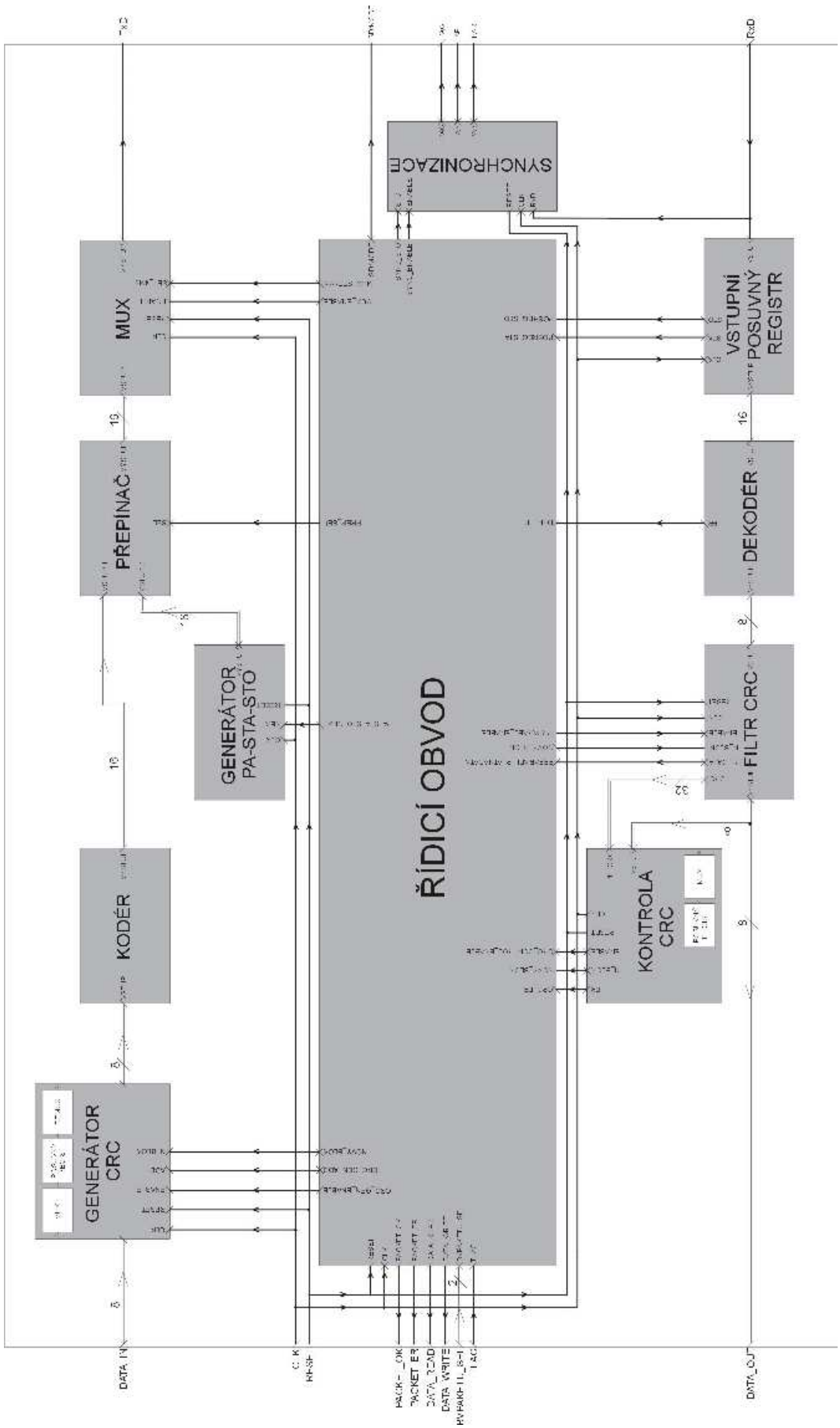
signál	vstup / výstup I / O	popis signálu
TxD	O	vysílaná data (Transmit Data)
RxD	I	přijímaná data (Receive Data)
SD/MODE	O	vypni / režim (Shut Down/Mode)

6.2 Blokové schéma obvodu, popis jednotlivých bloků

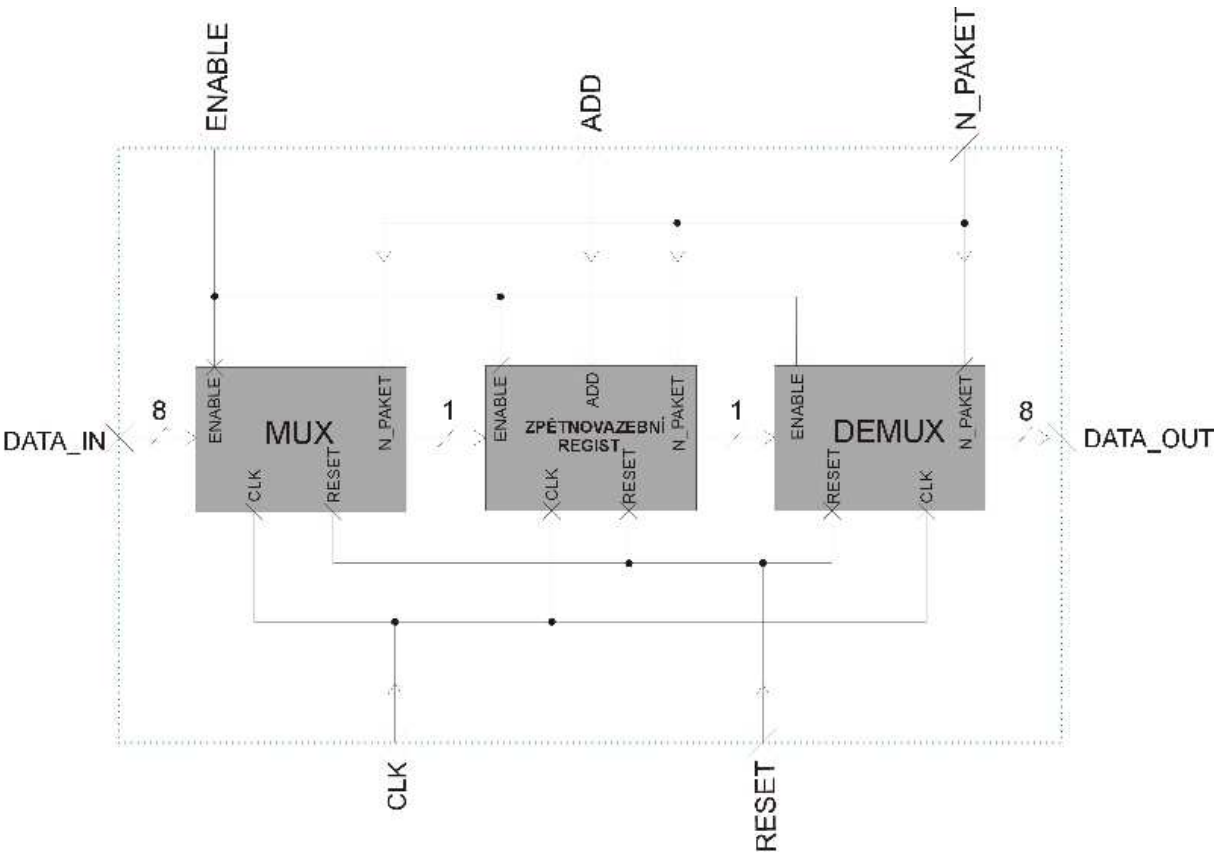
Blokové schéma (obr. č. 9) řeší rozdělení obvodu do několika celků, které zajišťují základní funkce komunikačního obvodu. Jednotlivé bloky mezi sebou komunikují pomocí datových a řídicích signálů. Ve vysílací části blokového schématu bloky transformují vstupní 8-bitová paralelní data do sériového toku dat (paketu) tak, jak jej definuje komunikační protokol. Přijímací část opačně ze sériového toku dat získává 8-bitová data očištěná o režii paketu. Řídicí signály jsou vysílány nebo detekovány řídicím odvodem, který podle nich upravuje a synchronizuje činnost ostatních (podřízených) bloků.

6.2.1 Generátor CRC

Generátor CRC (obr. č. 10) zajišťuje generování 32-bitové posloupnosti CRC, která slouží k detekci chyb. Vygenerované 4 bajty CRC kódu se připojí za datové bajty a dále se s nimi provádějí stejné operace jako s daty (kódování 4PPM modulací). Na vstupu bloku **Generátor CRC** je dělička frekvence hodinového signálu, protože v následných blocích



Obr. 9 – Blokové schéma obvodu



Obr. č. 10 – Blokové schéma Generátoru CRC

jsou již data zakódována pomocí 4PPM modulace. Po zakódování 4PPM modulací je potřeba hodinový signál o dvojnásobné frekvenci (8 MHz).

Blok Generátor CRC se skládá z multiplexoru, zpětnovazebního registru a demultiplexoru. Multiplexor slouží k převedení paralelního toku dat na sériový, který vstupuje do posuvného registru. Ve zpětnovazebním registru jsou zavedeny zpětné vazby podle polynomu $CRC(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$

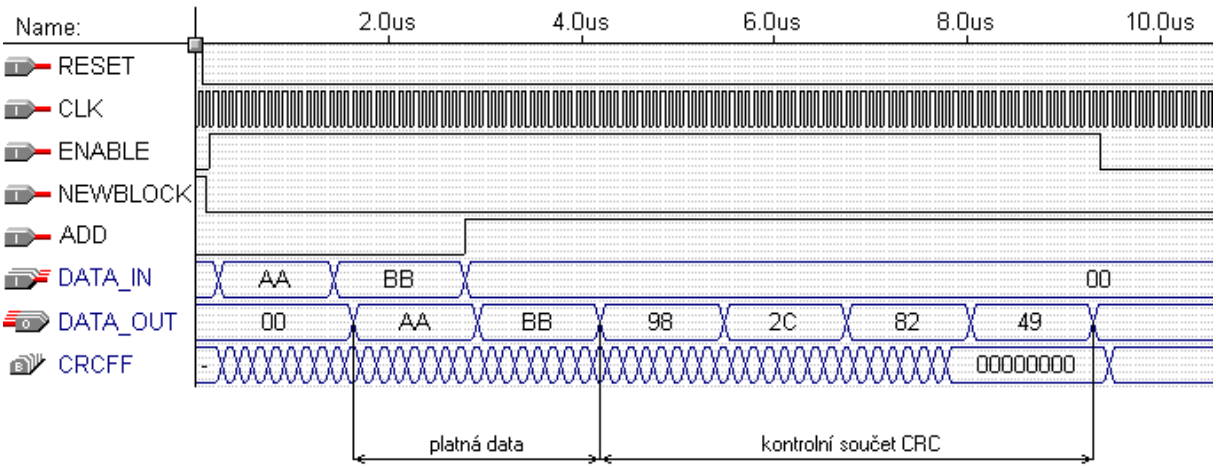
Tento polynom definuje norma IEEE 802 CRC32. Po vstupu všech datových bitů se v zpětnovazebním registru vytvoří 32-bitová posloupnost CRC. Za zpětnovazebním registrem následuje demultiplexor, který převádí sériový tok dat zpět na paralelní.

Řídicím signálem *ENABLE* se blokuje činnost Generátoru CRC.

Řídicí signál *ADD* slouží k přivedení vygenerované posloupnosti CRC na výstup ze zpětnovazebního registru. Vysílá se negace vygenerovaného CRC součtu.

Řídicí signál *N_PAKET* je využíván k přednastavení zpětnovazebního registru na hodnotu H'FF' a k vynulování čítačů. Posuvný registr se na tuto hodnotu musí nastavit při každém novém výpočtu kontrolního CRC součtu.

Průběh signálů v obvodu Generátor CRC je na obrázku 11.



Obrázek č. 11 – Průběh signálů na bloku Generátor CRC

6.2.2 Kodér

Kodér slouží k zakódování vstupního bajtu do šestnáctibitového slova v kódu 4PPM. Způsob, jak se kódují data, je uveden v teoretické části 4.1 popisující fyzickou vrstvu.

Kodér je čistě kombinační obvod, proto nemá žádný hodinový nebo resetovací vstup.

6.2.3 Generátor PA, STA a STO

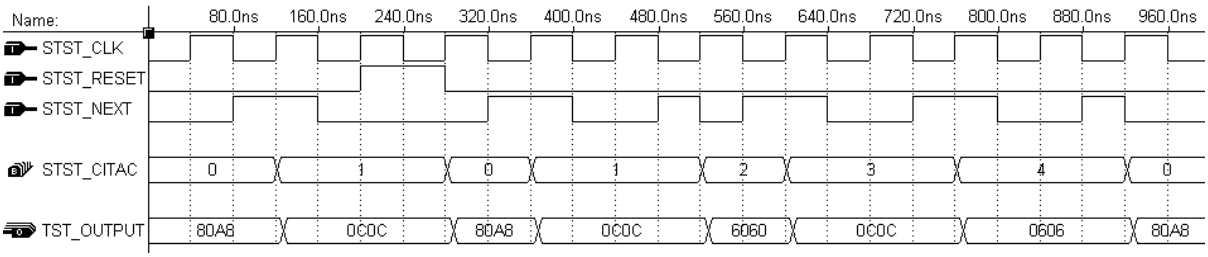
Blok generátor generuje konstantní posloupnosti bitů v paketu. Jsou to tyto části paketu: PA (*Preamble Field*), STA (*Start Flag*) a STO (*Stop Flag*). Jedná se o konstanty, které jsou na výstupu cyklicky generovány v závislosti na vstupním řídicím a hodinovém signálu.

Hodinový vstup *CLK* synchronizuje okamžik cyklické změny konstanty s ostatními částmi obvodu, vstup *STST_NEXT* udává, při které náběžné hraně hodinového signálu má k této změně dojít.

Vstup *STST_RESET* uvede obvod kdykoliv do výchozího stavu, tedy do stavu, kdy je na výstupu obvodu sekvence PA.

Protože sekvence bitů STA a STO jsou 32-bitové a datová sběrnice je v této části obvodu 16-bitová, jsou posloupnosti STA i STO rozděleny na dvě šestnáctibitové posloupnosti bitů, z nichž první část je pro obě dvě posloupnosti shodná.

Průběh signálů na tomto bloku je na obrázku 12.



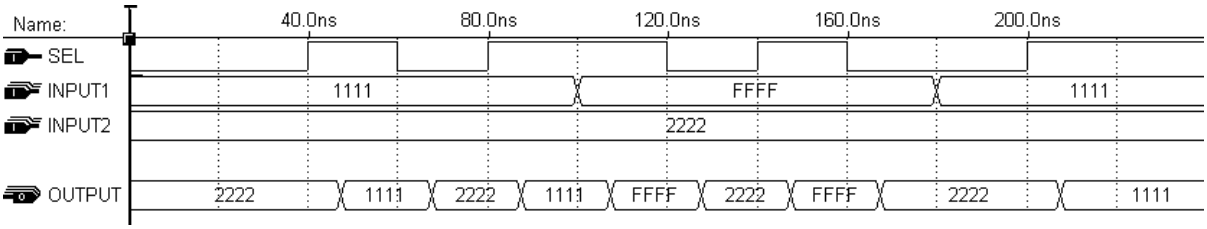
Obr. 12 – Průběh signálů na bloku Generátor PA, STA, STO

6.2.4 Přepínač jednoho z více vstupů na výstup

Tento blok slouží k přenosu jednoho z více (v naší aplikaci jeden ze dvou) šestnáctibitových paralelních vstupních signálů na šestnáctibitový paralelní výstup v závislosti na řídicím signálu *SEL*. Nastavení výstupu není závislé na hraně, nýbrž na úrovni tohoto signálu (každé z logických úrovní vstupu *SEL* odpovídá připojení jiného vstupu na výstup). Jedná se tedy o logickou výhybku, kdy se pomocí vstupního signálu přepíná mezi dvěma výstupy. Obvod je tvořen pouze kombinační logikou.

Obvod je použit k vytváření paketu. Na jeden vstup je přiváděna datová část paketu DD, na druhém vstupu jsou konstanty generované blokem **Generátor PA, STA a STO**. Na začátku vysílání paketu je na výstup přepojena větev obsahující konstanty PA a později STA. Po jejich odvysílání se přepne přepínač do druhé větve a odvysílá se část paketu DD. Po odvysílání DD se přepínač přepne zpět a dokončí se paket sekvencí STO.

Průběh signálů na tomto bloku v závislosti na vstupech je na obrázku 13.



Obr. 13 – Průběh signálů na funkčním bloku Přepínač.

6.2.5 Multiplexor

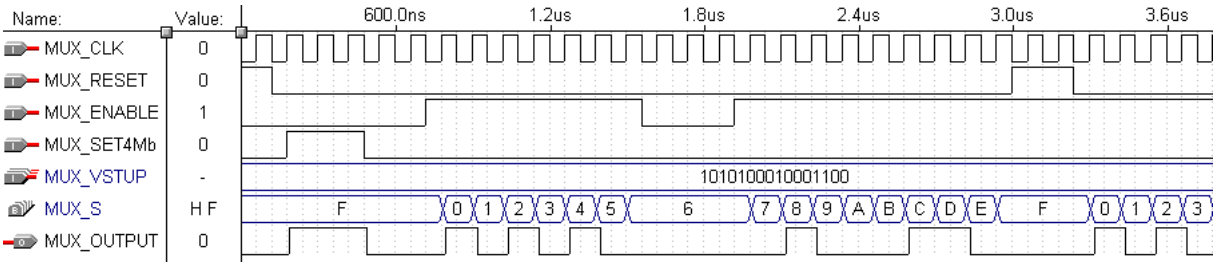
Blok slouží k převedení šestnáctibitového paralelního vstupu (již kompletní paket v šestnáctibitovém paralelním tvaru) do sériového výstupu. Obvod je synchronizován hodinovým vstupem *CLK*. K řízení je slouží signály *MUX_ENABLE* a *MUX_RESET* a *MUX_SET_4Mb*.

Signál *MUX_ENABLE* musí být v logické jedničce, aby se na výstup přenášela data z odpovídajícího vstupu a aby se při náběžné hraně hodinového signálu přepnul vnitřní ukazatel na další vstup.

Vynulování vnitřního ukazatele se provádí pomocí pulsu signálu *MUX_RESET*.

Na řídicí signál *MUX_SET4Mb* obvod reaguje asynchronně. Slouží k uvedení přijímače a vysílače infračervených pulsů do stavu, kdy je schopen přijímat a vysílat data rychlostí 4Mb/s.

Průběhy signálů na multiplexoru jsou na obrázku 14.



Obr. 14 – Průběhy signálů na bloku Multiplexor.

6.2.6 Synchronizace hodinového signálu přijímače s vysílačem

K synchronizaci hodinového signálu se využívá napěťově řízeného oscilátoru s fázovým závěsem.

Tento blok má za úkol řídit oscilátor s fázovým závěsem. Pomocí analogového přepínače se přepíná mezi signály, kterými je řízen napěťový oscilátor. Napětí, které řídí napěťový oscilátor, je ve stavu vysílání generováno odporovým děličem tak, aby se dosáhlo frekvence 8 MHz. Ve stavu přijímání se napětí generuje pomocí fázového závěsu.

K řízení analogového přepínače slouží výstup *Sync_AP*.

Na výstupu *Sync_PA* je synchronizační směs PA generovaná pomocí hodinového signálu přijímače.

Na výstupu *Sync_PAR* je přijatá synchronizační směs PA, kterou začíná každý paket přenosu (viz kapitola 1.4). Je to stejný signál jako RxD, ale očištěný o části paketu STA, DD, STO.

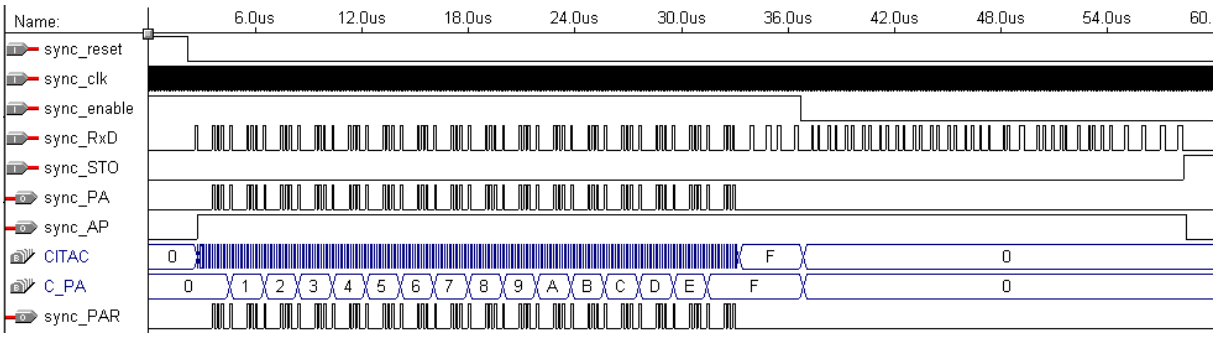
Výstupy *PAG* a *PAR* vstupují do fázového komparátoru a nastavují takovou úroveň napětí, aby napěťový oscilátor generoval stejnou frekvenci, na jaké je nastaven oscilátor vysílače.

Vstup *Sync_RxD* jsou data, která přichází z infračerveného přijímače a na základě kterých se začnou generovat výstupy *PAG* a *PAR*.

Vstup *ENABLE* povoluje generování PA sekvence na výstupu *PAG*.

Vstup *Sync_STO* určuje, kdy se má přepnout výstup *AP* zpět do log. nuly. Je to po přijetí sekvence STO, která ukončuje každý paket.

Průběhy na tomto bloku jsou zobrazeny na obrázku č. 15.



Obr. 15 – Průběhy signálů na bloku Synchronizace hodinového signálu

6.2.7 Vstupní posuvný registr

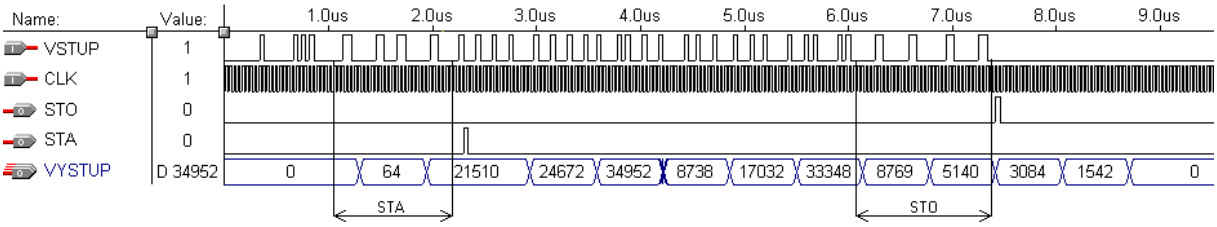
Vstupní posuvný registr slouží k převodu sériových dat ze vstupu na šestnáctibitový paralelní výstup.

Vstupní posuvný registr je 32-bitový, protože musí být schopen rozpoznávat 32-bitové sekvence STA a STO. Výstup je tvořen 16. až 31. bitem, vstup je zaveden do nultého bitu posuvného registru.

Výstup *STA* je v log. jedničce, je-li ve vstupním posuvném registru sekvence STA a výstup *STO*, je-li v posuvném registru posloupnost STO.

Synchronizace obvodu je zajišťována hodinovým vstupem *CLK*. Blokování činnosti obvodu signálem *ENABLE* není třeba. Obvod může pracovat stále, neboť jeho výstupy jsou dále zpracovávány a vyhodnocovány pouze v době definované řídicím obvodem.

Průběh signálů na vstupním posuvném registru v okamžiku nalezení posloupnosti STA (která nuluje čítač) zobrazuje obrázek 16.

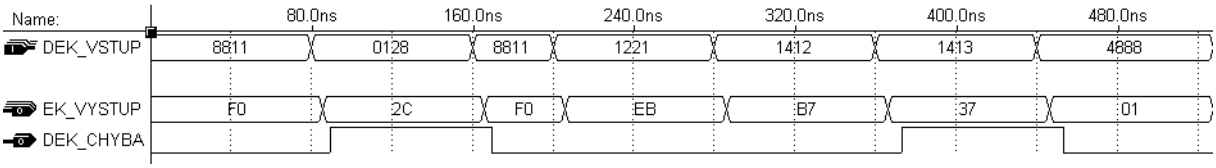


Obr. 16 – Průběhy signálů na bloku Vstupní posuvný registr

6.2.8 Dekodér

Dekodér je obvod pro dekódování vstupních šestnáctibitových dat v 4PPM kódu na výstupní osmibitová data. V případě, že vstupní šestnáctice bitů neodpovídá žádnému zakódovanému znaku (v každé čtveřici 0.-3., 4.-7., 8.-11. a 12.-15. bitu není právě jedna logická jednička), nastaví se výstupní řídicí bit *DEK_CHYBA*. Data na výstupu v tento okamžik nejsou platná. To nemá na činnost celého zařízení vliv, neboť tato data nejsou v době, kdy je signál *DEK_CHYBA* v logické jedničce, dále zpracovávána.

Průběh signálů na dekodéru pro správně přenesená i pro poškozená data (H‘0128‘ a H’1413‘) jsou na obrázku 17.



Obr. 17 – Průběhy signálů na bloku dekodér

6.2.9 Filtr CRC kódu

Tento obvod slouží k odfiltrování 32 bitového CRC kódu. Obvod zpožďuje data ve 4 x 8 bitovém registru ze vstupní 8 bitové datové sběrnice na výstupní 8 bitovou sběrnici.

Na výstupu *CRC* je 32 bitů vnitřního registru. Výstup využívá obvod **Kontrola CRC** ke kontrole přijatého paketu. 32 bitů CRC kódu je ve vnitřním registru právě v době, kdy je přijata 32-bitová posloupnost STO.

Výstup *PLATNADATA* informuje řídicí obvod o platných datech na výstupu.

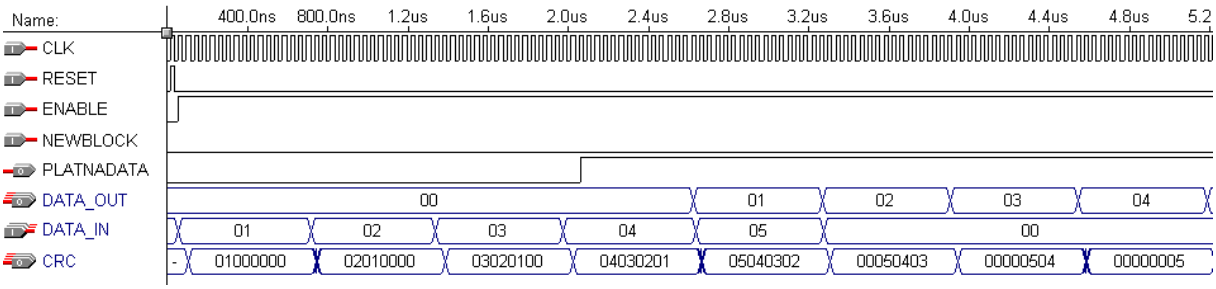
Vstup *CLK* synchronizuje výstup s ostatními obvody.

Vstup *RESET* slouží k uvedení celého obvodu do počátečního stavu (vynulují se čítače a vnitřní posuvný registr).

Vstup *NEWBLOK* je řídicím signálem generovaný řídicím obvodem a uvádí tento obvod do počátečního stavu vždy před přijetím nového paketu.

Vstup *ENABLE* povoluje obvod.

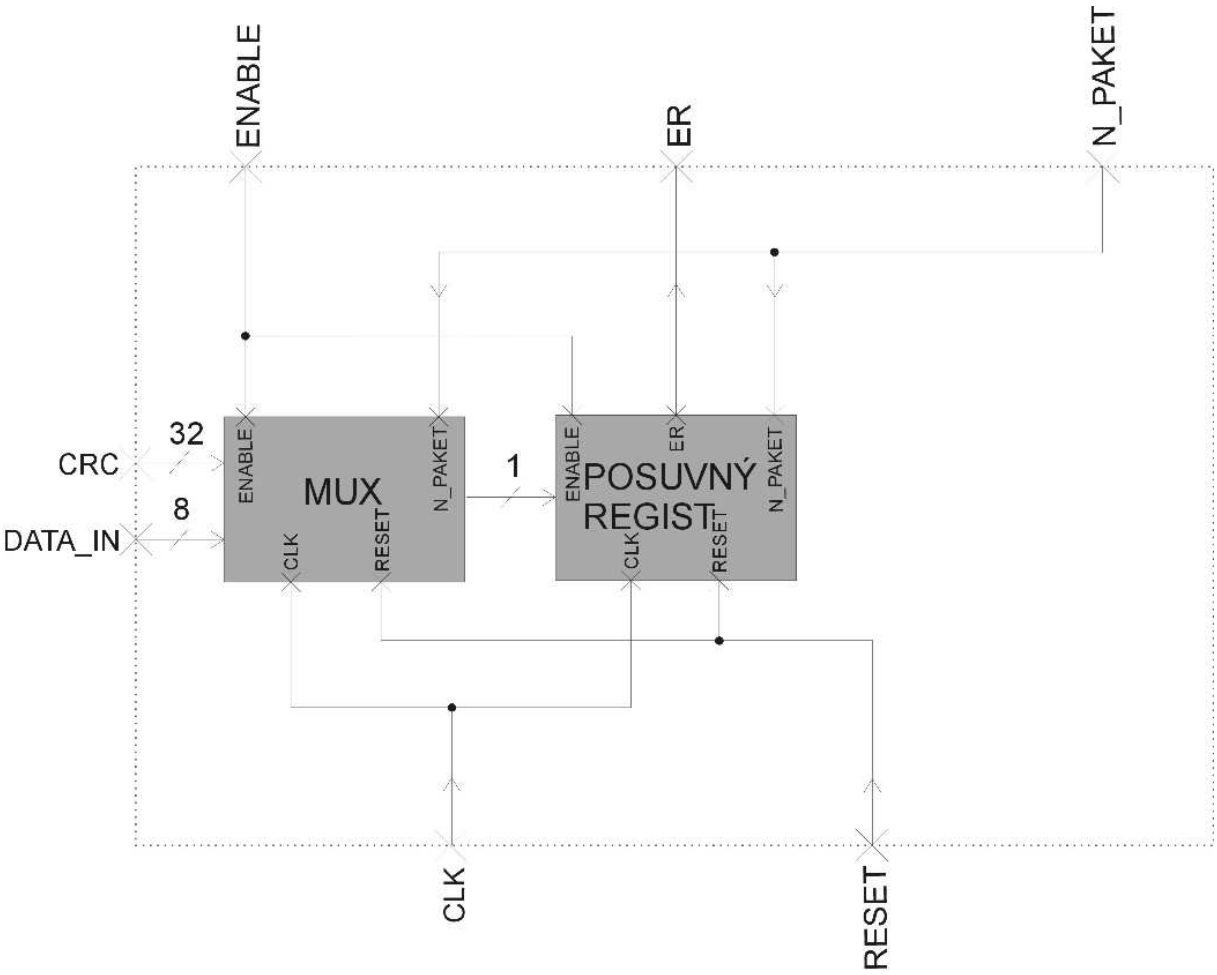
Na obrázku 18 jsou zobrazeny výstupy, reagující na vstupní signály.



Obr. 18 – Průběhy signálů na bloku Filtr CRC kódu

6.2.10 Kontrola CRC

Blok **Kontrola CRC** (viz obr č. 19) slouží ke kontrole přijatého paketu. Blok pracuje stejně jako blok **Generátor CRC**. Obsahuje stejný multiplexor a zpětnovazební registr, který generuje 32-bitovou posloupnost bitů. Ta je po vstupu posledního datového bajtu zkontrolována s přijatou CRC posloupností.



Obr. 19 – Blokové schéma bloku Kontrola CRC

O výsledku kontroly je informován **Řídicí obvod** výstupním signálem *ERR*.

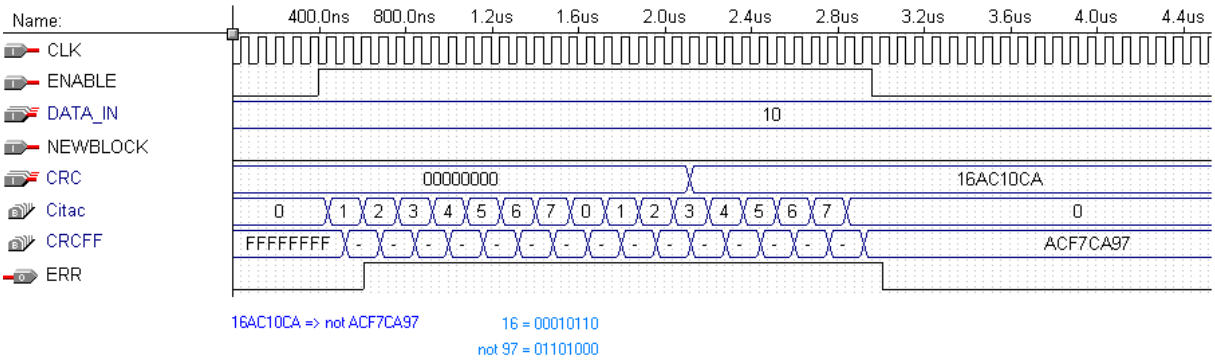
Vstupní signál *ENABLE* povoluje činnost celého bloku. Pokud je tento signál v logické jedničce, pak celý obvod reaguje synchronně na náběžné hrany vstupního hodinového signálu *CLK*.

Reset je asynchronní nulování celého bloku. Blok se dostane do počátečního stavu.

NEWBLOK je vstupní signál, který synchronně uvede blok do počátečního stavu před každým novým příjmem paketu. Nastaví vstupní posuvný registr na výchozí hodnotu H'FF' a vynuluje všechny čítače.

Na vstupu *CRC* je hodnota CRC kódu, která byla přijata jako kontrolní součást paketu. Tento vstup je stále porovnáván se stavem vnitřního posuvného registru a v případě, že se jejich hodnoty rovnají, pak je výstup *ERR* nastaven do logické jedničky.

Průběh signálů na bloku Kontrola CRC je zobrazen na obrázku 20. Vygenerovaný CRC kód je ve vnitřním posuvném registru *CRCFF*. Přijatý kód CRC je na vstupu *CRC*.



Obr. 20 – Průběh signálů na bloku Kontrola CRC

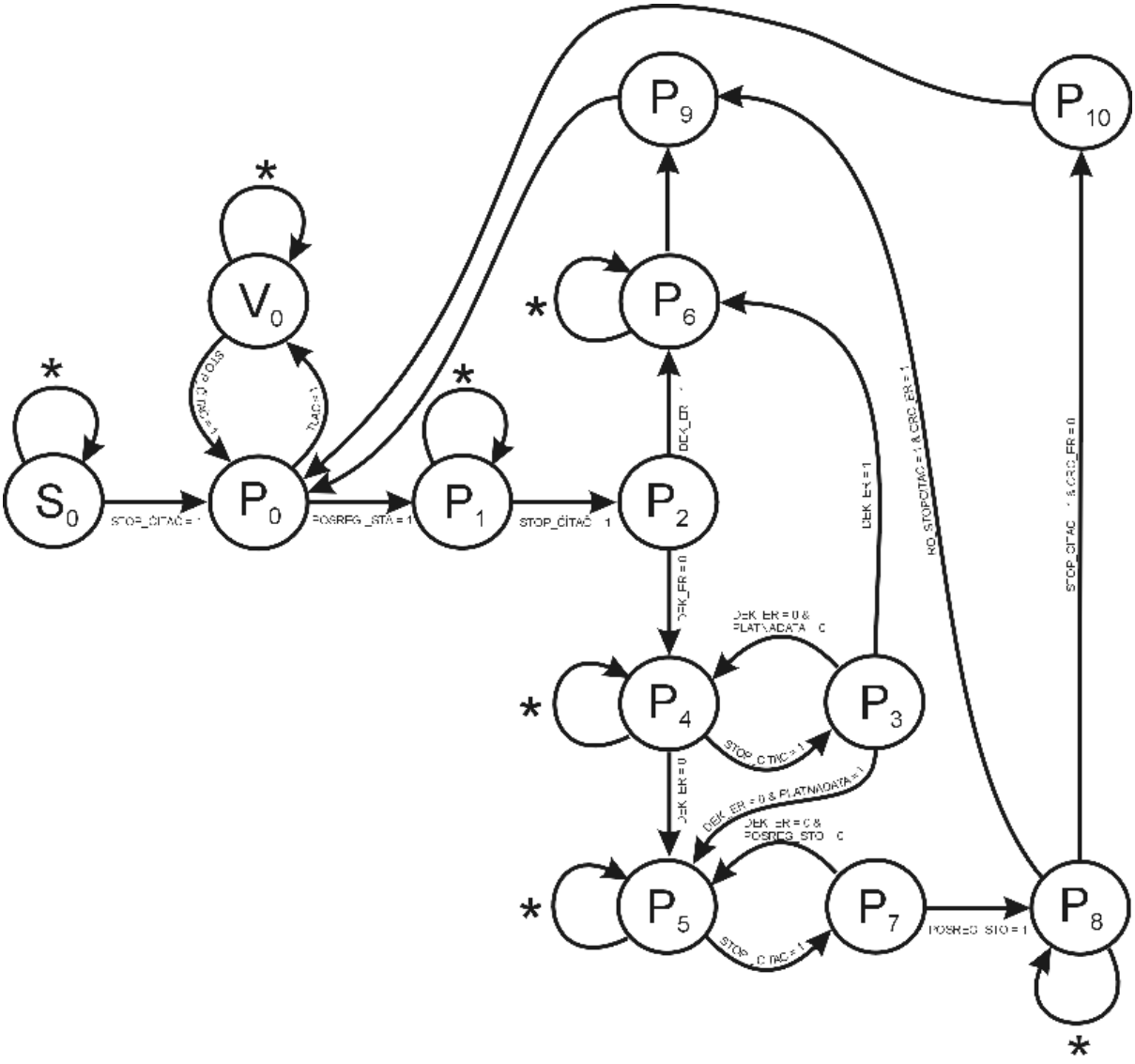
6.2.11 Řídicí obvod

Úkolem řídicího obvodu je koordinovat činnost všech ostatních bloků tak, aby byl zajištěn celý přenos dat. Řídicí obvod vysílá do ostatních bloků výše popsané signály *CRC_GEN_ENABLE, CRC_GEN_ADD, NOVY_BLOK, STST_NEXT, PREP_SEL, MUX_ENABLE, MUX_SET4Mb, SYNC_STO, SYNC_ENABLE, FILTR_ENABLE, CRC_CONTROL_ENABLE* a čte z bloků hodnoty řídicích signálů *POSREG_STA, POSREG_STO, FILTR_PLATNADATA, CRC_ER a DEK_ER*. Řídicí obvod komunikuje s vyšší spojovou vrstvou pomocí vstupů *TLAC* (signál pro vyslání jednoho kompletního paketu), *RESET* (tlačítko pro nucený návrat do výchozího stavu), *BVPAKETU_SEL* (2-bitový vstup, který určuje počet vysílaných bajtů v paketu) a výstupů *PAKET_OK, PAKET_ER,*

DATA_READ, DATA_WRITE. Činnost obvodu je synchronizována hodinovým signálem *CLK*.

Řídicí obvod je vytvořen na bázi Moorova stavového automatu. Řídicí obvod je rozšířen o čítač, který zabezpečí výraznou redukci počtu stavů. U klasického Moorova automatu jsou výstupy definované pouze stavem automatu. V případě použití čítače závisí výstupy automatu i na stavu čítače. To umožňuje např. vyslání paketu v jednom stavu automatu. Další výhoda je v tom, že lze např. vyslání jednoho paketu vyřešit pro libovolný počet bajtů v jednom paketu (určeno pouze jednou snadno změnitelnou konstantou). Schéma automatu je na obrázku 21.

Stav ‘S_0‘ slouží k inicializaci přijímače a vysílače IRMS6400. Stav ´V_0´ slouží k vyslání jednoho kompletního paketu a stavy označené písmenem ´P´ označují přijímač.



Obr. 21 – Návrh Moorova stavového automatu pro řídicí obvod fyzické vrstvy

* – Hrany označené tímto symbolem naznačují, že řídicí obvod zůstává ve stejném stavu, pokud není splněna některá z podmínek pro přechod do dalšího stavu.

Stav S_0

Stav S_0 je výchozím stavem automatu. Automat se do stavu S_0 dostane po resetu obvodu, tedy tehdy, pokud se na vstup do řídícího obvodu *Reset* přivede logická jednička.

Úkolem stavu S_0 je nastavit integrovaný obvod vysílače a přijímače IRMS6400 do stavu, kdy je schopen přijímat a vysílat rychlostí 4 Mb/s. Obvod ve stavu S_0 zůstává po dobu pevně danou rozsahem čítače pro tento stav. Za tuto dobu přivede na výstupy *MUX_SET4Mb* a *SD/MODE* takové signály, aby obvod IRMS6400 přijímal a vysílal na 4 Mb/s.

Stav P_0

Ve stavu P_0 se čeká na pokyn vyšší spojové vrstvy na vstupu *TLAC* o vyslání jednoho paketu dat (přechod do stavu V_0) nebo na příchod sekvence STA (přechod do stavu P_1). Jinak automat setrvává ve stavu P_0.

Stav P_1

Tento stav zajišťuje, že po nalezení sekvence STA na vstupu automat počká přesně definovanou dobu, než se v posuvném registru dostanou na výstup platná data. Další stav je P_2 a přechází se do něj v případě, že je na čítači konstanta, která definuje dobu čekání automatu v tomto stavu, jinak se setrvává ve stavu P_1.

Stav P_2

Do stavu P_2 se řídicí obvod dostane ze stavu P_1 v případě, že je na výstupu z posuvného registru první zakódované slovo paketu. Podle toho, zda v tomto slově nastala chyba, je další stav P_4 (chyba nenastala) nebo P_6 (chyba nastala). Do dalšího stavu se tedy přechází okamžitě, čítač není nevyužíván.

Stav P_3

Stav P_3 má podobnou funkci jako stav P_2. Obvod se do tohoto stavu dostane ze stavu P_4 po dočítání vnitřního čítače řídícího obvodu. Řídicí obvod ve stavu P_3 kontroluje zda v dekódovaném slově nastala chyba. V případě chyby je dalším stav P_6 v opačném případě P_4 nebo P_5. Do stavu P_5 se obvod dostane pokud na výstupu z bloku **Filtr CRC kódu** jsou platná data. V opačném případě se dostane obvod do stavu P_4. Do dalšího stavu se přechází okamžitě, čítač není nevyužíván.

Stav P_4

Ve stavu P_4 se čeká 15 hodinových impulsů na další šestnáctici bitů (další bajt zakódovaný 4PPM kódem) na výstupu z posuvného registru. Po uplynutí této doby je vždy následující stav P_3.

Stav P_5

Ve stavu P_5 se čeká (tak jako ve stavu P_4) 15 hodinových impulsů na další šestnáctici bitů na výstupu z posuvného registru. Po uplynutí této doby je vždy následující stav P_7. Na rozdíl od stavu P_4 je v tomto stavu povolena činnost bloku **Kontrola CRC** výstupem *CRC_CONTROL_ENABLE*.

Stav P_6

Do stavu P_6 se automat dostane ze stavu P_2, P_3 nebo P_7 v případě, že dekodér detekoval chybu v přenosu. Čeká se zde na přijetí sekvence STO ukončující paket. Po přijetí této posloupnosti bitů dojde k uvolnění vysílací cesty pro vyslání odpovědi. Pak automat přechází do stavu P_9.

Stav P_7

Stav P_7 má podobnou funkci jako stav P_2 nebo P_3. Obvod se do tohoto stavu dostane ze stavu P_5 po dočítání čítače. Řídicí obvod ve stavu P_7 kontroluje zda v dekódovaném slově nastala chyba. V případě chyby je dalším stav P_6 v opačném případě P_5. Pokud je detekována sekvence STO je další stav P_8. Do dalšího stavu se přechází okamžitě, čítač není nevyužíván.

Stav P_8

Ve stavu P_8 se kontroluje zda proběhla úspěšně kontrola přijatého paketu pomocí CRC kódu. Podle toho, zda byla detekována chyba pomocí CRC kódu, je dalším stavem obvodu P_{10} (chyba nenastala) nebo P_9 (chyba nastala).

Stav P_9

Stav P_9 informuje vyšší spojovou vrstvu o chybně přijatém paketu. Dalším stavem je výchozí stav P_0.

Stav P_{10}

Stav P_{10} informuje vyšší spojovou vrstvu o správně přijatém paketu. Dalším stavem je výchozí stav P_0.

Stav V_0

Stav V_0 slouží k vyslání jednoho kompletního paketu. Důležitou součástí tohoto stavu je čítač. Na základě stavu čítače se mění výstupy řídicího obvodu. Pomocí stavů čítače se tedy řídí vyslání celého jednoho paketu. Dalším stavem je stav P_0, do kterého se obvod dostane po dočítání čítače.

7. Návrh a realizace spojové vrstvy

Spojová vrstva komunikačního protokolu musí zajistit rychlé a správné navázání a ukončení komunikace a řízení toku dat po rozhraní.

Při návrhu spojové vrstvy komunikační jednotky jsou uvažována tato zjednodušení a zjednodušující předpoklady:

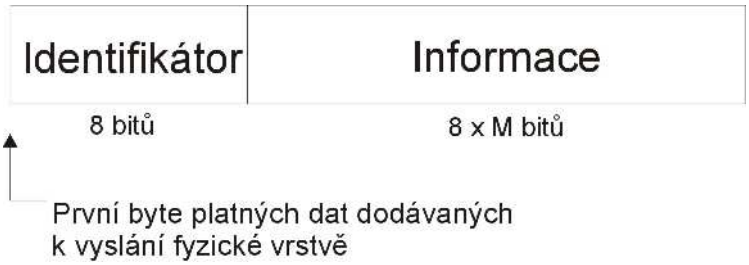
1. Navazování komunikace a přenos dat při jediné přenosové rychlosti 4 Mb/s.
2. Ve vzájemném dosahu jsou vždy pouze dvě komunikační zařízení.
3. Přenos dat probíhá pouze jedním směrem.

Navazování a přenos dat stejnou přenosovou rychlostí zjednoduší návrh o implementaci další fyzické vrstvy pro rychlost 9,6 kb/s. Předpoklad, že jsou v dosahu vždy pouze dvě zařízení umožní výrazně zrychlit navazování komunikace. Zjednoduší se rámce pro spojovou vrstvu, protože nebude třeba vysílat s každým rámcem adresové pole. Přenos dat jedním směrem neznamená, že jedna z komunikačních stanic nebude vůbec vysílat. Stanice, která bude přijímat data, musí být schopná potvrdit vysílající stanici správně přijatý rámec. Tato stanice také bude iniciátorem navázání komunikace. Přijímající stanice tedy bude vysílat pouze řídicí rámce, kterými bude žádat o přenos dat, potvrzovat správně či chybně přijaté rámce a ukončovat komunikaci.

Motivací pro vytvoření tohoto rozhraní byl přenos dat z nedostupného místa, kde se data budou čerpat ze senzoru do pevné jednotky a dále přenášet po infračerveném rozhraní do mobilní jednotky. Návrh spojové vrstvy komunikačního rozhraní je tedy vytvářen jednak ze strany jednotky senzoru (JS) a jednak mobilní jednotky (MJ).

7.1 Struktura přenosového rámce MJ a JS

Rámce MJ a JS se dělí do dvou polí. Osmibitové pole Identifikátor obsahuje zprávu o typu rámce a v případě řídícího rámce i některé další informace. Pole Informace obsahuje platná data přenosu. Struktura rámce je zobrazena na obr. 22.



Obr. 22 – Struktura rámce MJ a JS

Řídicí a datové rámce MJ a JS:

1. Navázání komunikace - vysílá MJ
   Identifikátor - B’10101010’
   Informace - -
2. Datový rámec - vysílá JS
   Identifikátor - B’00000000’
   Informace - M x 8 bitů
3. Odpověď na správně přijatý rámec - vysílá MJ i JS
   Identifikátor - B’10000000’
   Informace - -
4. Odpověď na chybně přijatý rámec - vysílá MJ i JS
   Identifikátor - B’10111111’
   Informace - -
5. Ukončení komunikace - vysílá JS
   Identifikátor - B’11111111’
   Informace - 1 x 8 bitů (adresa JS)

Řídicí rámec od datového je rozlišen sedmým bitem osmibitového pole identifikátor viz obr. 23.

Obr. 23 – Struktura pole Identifikátor

7.2 Navázání komunikace, přenos dat a ukončení komunikace mezi MJ a JS

Navázání komunikace zajišťuje MJ. MJ vyšle řídící paket **Navázání komunikace**, na který zareaguje JS tím, že začne vysílat datové rámce. Po odvysílání každého datového rámce JS počká na odpověď od MJ v podobě řídícího rámce ve tvaru **Odpověď na správně přijatý rámec** nebo **Odpověď na chybně přijatý rámec.** Pokud JS dostane od vyšší vrstvy příkaz k ukončení přenosu dat, pak vyšle ukončující řídící rámec **Ukončení komunikace**. Poslední rámec přenosu vyšle MJ ve tvaru **Odpovědi na správně přijatý rámec.**

7.3 Stavový automat MJ a JS

Mobilní jednotka a jednotka senzoru jsou tvořeny stavovým automatem, který řídí vysílání rámců tak, jak je definováno v kapitole 7.2. Tento stavový automat je automatem Moorova typu, který je rozšířen o vnitřní čítač.

Vnitřní čítač je využívám především k omezení doby, po kterou automat zůstává v některých stavech. Tím se chrání automat před zastavením ve stavu ve chvíli, kdy dojde k jedné z nepravděpodobných chyb v přenosu (chyby v sekvencích STA a STO datových rámců, chyby v řídících rámcích). Dočítáním čítače se automat vrátí do výchozího stavu. Přenos dat pak musí být spuštěn znovu.

Jednotlivé rámce jsou vysílány fyzickou vrstvou. Stavový automat MJ a JS řídí fyzickou vrstvu pomocí výstupů a tím ovlivňuje strukturu rámců. Na základě vstupů z fyzické vrstvy a vstupů z vrstvy vyšší automat přechází mezi stavy a tím řídí celou komunikaci. Vnitřní vstupy a výstupy, které slouží pro komunikaci s fyzickou vrstvou, jsou shodné se vstupy a výstupy definovanými v kapitole 6.1.

7.4 Návrh mobilní jednotky (MJ)

Mobilní jednotka nevysílá žádné datové rámce. Řídí tok dat po infračerveném rozhraní pomocí řídících rámců, které navazují komunikaci, potvrzují správně a chybně přijaté rámce a ukončují komunikaci.

7.4.1 Vstupy a výstupy mobilní jednotky

signál	vstup / výstup I / O	popis signálu
START	I	start přenosu dat
DATA_OUT	O	výstupní 8 bitová datová sběrnice
CLK	I	hodinový signál
RESET	I	resetovací signál
TR_OK	O	celý přenos proběhl korektně
TR_ER	O	při přenosu nastala chyba
DATA_WRITE	O	platná data na DATA_OUT
ADR_DEC	O	chyba v posledním rámci (rámec bude poslán znovu)

7.4.2 Stavový automat mobilní jednotky

Stavový automat spojové vrstvy mobilní jednotky obsahuje 15 stavů (viz obr. 24), které jsou definovány takto:

Stav S_0

V tomto stavu čeká automat na signál START z vyšší vrstvy (přechod do stavu S_1)., jinak zůstává automat ve stavu S_0. Reakcí na signál START je začátek přenosu dat.

Stav S_1

Stav S_1 slouží k vyslání prvního řídícího rámce přenosu dat. V tomto stavu se na vstup fyzické vrstvy TLAC přivede logická jednička a tím začne vysílání rámce. Ve stavu S_1 automat setrvává až do příchodu signálu DATAREAD od fyzické vrstvy, kterým fyzická vrstva žádá o první bajt přenášených dat na vstup DATA_IN.

Stav S_2

Ve stavu S_2 se čeká na signál DATAWRITE, kterým informuje fyzická vrstva spojovou vrstvu, že byl přijat první bajt paketu (další stav je S_3). V případě, že vnitřní čítač stavového automatu dočítá na hodnotu konstanty SX_STOP, pak je dalším stavem S_{11} (přenos bude přerušen – jednotka senzoru není v dosahu).

Stav S_3

Stav S_3 slouží k určení typu rámce. Automat zde kontroluje zda se jedná o datový a nebo řídící rámec. V případě datového rámce je další stav S_4 a v případě řídicího rámce je další stav S_5. Do dalšího stavu se přechází okamžitě (čítač není využíván).

Stav S_4

Ve stavu S_4 řídicí obvod čeká na konec rámce, který spojové vrstvě oznámí fyzická vrstva pomocí signálu RO_OK (další stav je S_7) nebo RO_ER (další stav je S_8). Zároveň se vyšší vrstvě předává signál DATAWRITE, který vyšší vrstvu informuje o novém bajtu dat na vstupu DATA_IN. V případě, že fyzická vrstva konec rámce nepřijme, po určité předem

definované době, dojde k přetečení vnitřního čítače řídicího obvodu a stavový automat se dostane do stavu S11 (přenos bude ukončen s chybou).

Stav S_5

Ve stavu S_5 obvod čeká na signál DATAWRITE, který oznamuje, že na vstupu DATA_IN se objeví adresa mobilní jednotky, která data vysílala.

Stav S_6

Ve stavu S_6 obvod čeká na konec ukončovacího rámce komunikace od jednotky senzoru. Příjem konce rámce oznámí fyzická vrstva na vstupech RO_OK nebo RO_ER podle toho, zda byl rámec přijat správně nebo chybně. V případě chybně přijatého rámce je další stav S_{12}. Pokud přijme fyzická vrstva rámec správně, pak je další stav S_{13}. V případě že MJ konec rámce vůbec nepřijme (nastala chyba v sekvenci STO), dojde po určité době k přetečení čítače řídicího obvodu a další stav je S_{11}.

Stav S_7

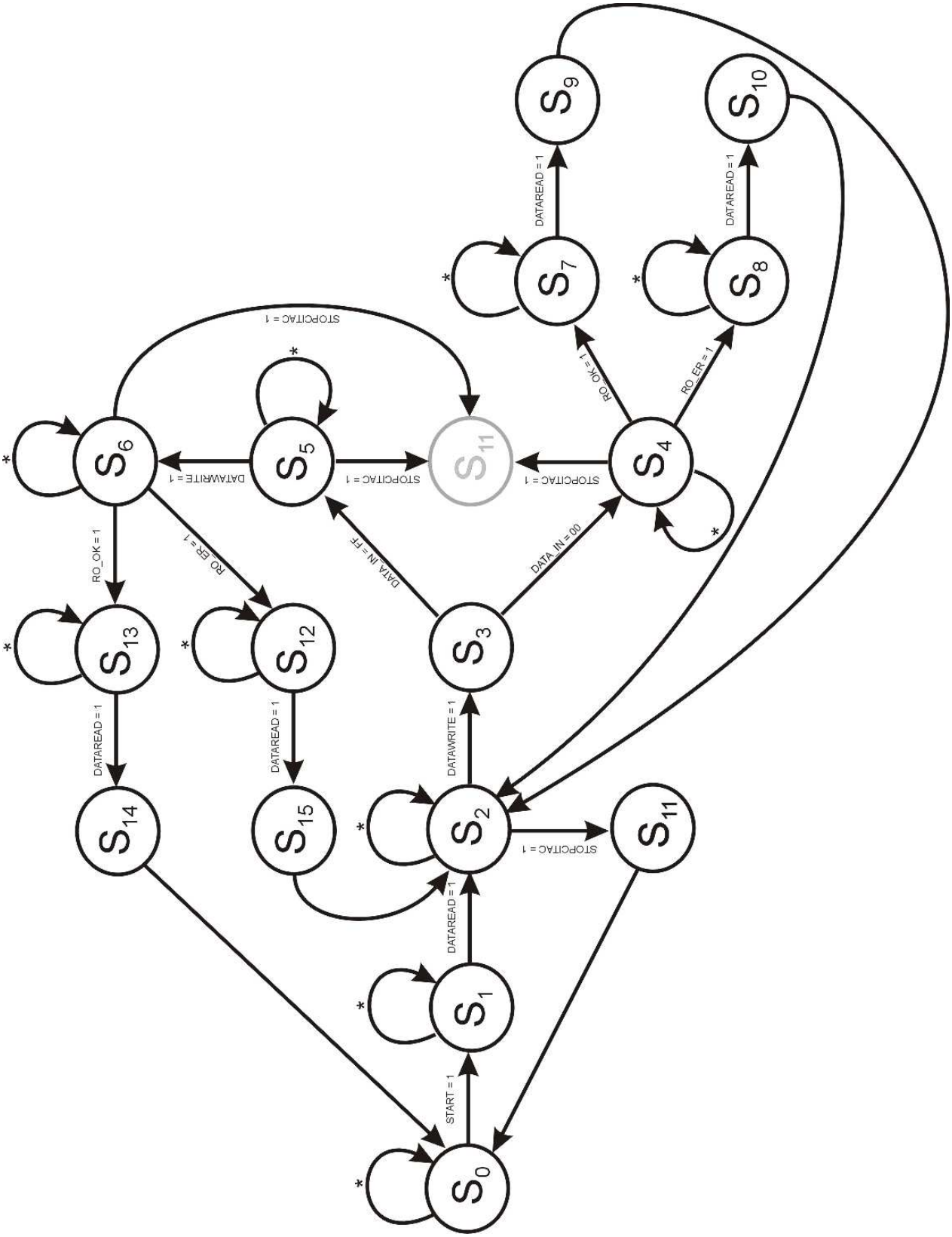
Stav S_7 vyšle řídicí rámec **Odpovědi na správně přijatý datový rámec**. Do dalšího stavu S_9 se přechází po žádosti (signál DATAREAD) fyzické vrstvy o vysílaná data na vstup fyzické vrstvy DATA_IN.

Stav S_8

Stav S_8 má podobnou funkci jako stav S_7. V tomto stavu se vysílá rámec **Odpověď na chybně přijatý rámec**. Do stavu S_{10} se přechází ve chvíli, kdy přijde signál DATAREAD, jinak obvod zůstává ve stavu S_8.

Stav S_9

Do dalšího stavu se obvod dostává okamžitě. Vnitřní čítač řídicího obvodu není využíván. Dalším stavem je stav S_2. Ve stavu S_9 se na vstup fyzické vrstvy DATA_IN přivede bajt, který je výše definován jako **Odpověď na správně přijatý rámec**.



Obr. 24 – Návrh Moorova stavového automatu pro řídicí obvod mobilní jednotky

* – Hrany označené tímto symbolem naznačují, že řídicí obvod zůstává ve stejném stavu, pokud není splněna některá z podmínek pro přechod do dalšího stavu.

Stav S_{10}

Stav S_{10} má podobnou funkci jako stav S_9. Na rozdíl od stavu S_9 se na vstup fyzické vrstvy DATA_IN přivede bajt definovaný jako **Odpověď na chybně přijatý rámec**.

Stav S_{11}

Ve stavu S_{11} není využíván vnitřní čítač řídicího obvodu. Řídicí obvod přechází do dalšího stavu okamžitě. V tomto stavu je na výstupu TR_ER logická jednička. Vyšší vrstva je tímto výstupem informována o tom, že přenos dat proběhl s chybou, a že musí být uskutečněn znovu. Dalším stavem je výchozí stav celého řídicího obvodu S_0.

Stav S_{12}

Do stavu S_{12} se obvod dostane, pokud došlo k chybě v posledním ukončujícím rámci přenosu. Ve stavu S_{12} se na vstup fyzické vrstvy TLAC přivede logická jednička a tím se vyšle rámec ve tvaru **Odpověď na chybně přijatý rámec**. Do dalšího stavu S_{15} se obvod dostane, pokud je na výstupu z fyzické vrstvy DATAREAD logická jednička.

Stav S_{13}

Do stavu S_{13} se řídicí obvod dostane, pokud byl poslední ukončující paket přenosu dat přijat v pořádku. V tomto stavu se tak jako ve stavu S_{12} na vstup fyzické vrstvy TLAC přivede logická jednička ta tím se začne vysílat rámec ve tvaru **Odpověď na správně přijatý rámec**. Do dalšího stavu S_{14} se obvod dostane, pokud je na výstupu z fyzické vrstvy DATAREAD logická jednička.

Stav S_{14}

Do dalšího stavu se obvod dostává okamžitě. Dalším stavem je stav S_0. Ve stavu S_{14} se na vstup fyzické vrstvy DATA_IN přivede hodnota H’80’, která je jednotkou senzoru identifikována jako **Odpověď na správně přijatý rámec**. Vnitřní čítač řídicího obvodu v tomto stavu není využíván.

Stav S_{15}

Ve stavu S_{15} se na vstup fyzické vrstvy DATA_IN přivede hodnota H'BF', která je jednotkou senzoru identifikována jako **Odpověď na chybně přijatý rámec**. Do dalšího stavu S_2 se obvod dostane okamžitě (vnitřní čítač řídicího obvodu není využíván).

7.5 Návrh jednotky senzoru (JS)

Jednotka senzoru vysílá datové rámce, ve kterých jsou přenášena platná data a jediný řídicí rámec, který slouží k ukončení přenosu. V řídicím rámci je zároveň obsažena i adresa jednotky senzoru (jeden bajt v poli Informace). Pokud odpověď na datový rámec přijde ve tvaru **Odpověď na chybně přijatý rámec**, jednotka senzoru informuje vyšší vrstvu a ta zajistí vyslání tohoto rámce znovu.

7.5.1 Vstupy a výstupy jednotky senzoru

signál	vstup / výstup I / O	popis signálu
DATA_IN	I	vstupní 8-bitová datová sběrnice
CLK	I	hodinový signál
RESET	I	resetovací signál
DATA_END	I	konec přenosu
TR_OK	O	celý přenos proběhl korektně
TR_ER	O	při přenosu nastala chyba
DATA_READ	O	platná data na DATA_IN
ADR_DEC	O	chyba v posledním rámci (rámec bude poslán znovu)

7.5.2 Stavový automat jednotky senzoru

Stavový automat spojové vrstvy jednotky senzoru obsahuje 18 stavů (viz obr. 25), které jsou definovány takto:

Stav S_0

Ve stavu S_0 řídicí obvod čeká na první rámec přenosu **Navázání komunikace** od MJ. Do dalšího stavu S_1 stavový automat přejde po příchodu signálu DATAWRITE, jinak zůstává ve stavu S_0.

Stav S_1

Ve stavu S_1 se kontroluje zda přijatý rámec je rámec **Navázání komunikace**. V případě, že na výstupu z fyzické vrstvy je hodnota H'AA', je další stav S_2, jinak se řídicí obvod vrací do stavu S_0. Do dalšího stavu se přechází okamžitě (vnitřní čítač není využíván).

Stav S_2

Ve stavu S_2 stavový automat čeká na konec rámce **Navázání komunikace**. Konec rámce ohlašuje fyzická vrstva výstupy RO_OK (další stav S_3) a RO_ER (další stav S_0). Pokud dojde k přetečení vnitřního čítače obvodu, je další stav S_4. Jinak zůstává obvod ve stavu S_2.

Stav S_3

Ve stavu S_3 JS informuje vyšší vrstvu o požadavku na přenos dat. Vyšší vrstva je informována pomocí výstupu DATAREAD z JS. Do stavu S_5 se obvod dostane okamžitě (čítač není využíván).

Stav S_4

Stav S_4 informuje vyšší vrstvu o chybě v komunikaci a dochází k resetu nižší fyzické vrstvy. Další stav je S_0. Obvod do stavu S_0 přechází okamžitě bez ohledu na vstupy a vnitřní čítač obvodu.

Stav S_5

Ve stavu S_5 se začínají vysílat datové rámce. Vstup fyzické vrstvy TLAC je nastaven do logické jedničky. Do dalšího stavu S_6 se obvod dostane po příchodu signálu DATAREAD fyzické vrstvy.

Stav S_6

Ve stavu S_6 se na vstup fyzické vrstvy DATA_IN přivede hodnota H'00', která identifikuje datový rámec. Do dalšího stavu se obvod dostane po příchodu dalšího impulsu na signálu DATAREAD.

Stav S_7

Ve stavu S_7 se vyšle zbytek datového rámce a čeká se na odpověď od MJ. Do dalšího stavu S_8 se obvod dostane po příchodu signálu DATAWRITE. Pokud dojde k přetečení vnitřního čítače obvodu je další stav S_4.

Stav S_8

Ve stavu S_8 je podle typu odpovědi na datový rámec (**Odpověď na správně přijatý rámec** nebo **Odpověď na chybně přijatý rámec**) další stav S_9 nebo S_{10}. V případě, že je přijat rámec v jiném tvaru než jak je uvedeno, je další stav S_4.

Stav S_9

Do stavu S_9 se obvod dostane v případě, že poslední datový rámec přijala MJ v pořádku. Obvod zde čeká na konec řídicího rámce odpovědi. Další stav je S_{11} je-li je signál fyzické vrstvy RO_OK = 1 (řídicí rámec byl přijat v pořádku) a vstup z vyšší vrstvy RO_END = 1 (žádost o ukončení přenosu dat). Do stavu S_5 se obvod dostane, pokud je signál fyzické vrstvy RO_OK = 1 a vstup z vyšší vrstvy RO_END = 0. Přenos dat bude ukončen chybou, je-li signál fyzické vrstvy RO_ER = 1 (další stav je S_4) nebo když dojde k přetečení vnitřního čítače obvodu. V ostatních případech zůstává obvod ve stavu S9.

Stav S_{10}

Do stavu S_{10} se obvod dostane v případě, že došlo k chybě při příjmu posledního datového rámce. Ve stavu S_{10} se čeká na konec řídicího rámce. Další stav je S_5, jestliže byl rámec odpovědi přijat správně (RO_OK = 1). Do stavu S_4 se obvod dostane pokud nastala chyba při příjmu řídicího rámce (RO_ER = 1) nebo dojde-li k přetečení vnitřního čítače obvodu. V ostatních případech zůstává obvod ve stavu S_{10}.

Stav S_{11}

Ve stavu S_{11} se na vstup fyzické vrstvy TLAC přivede logická jednička a spustí se vyslání rámce **Ukončení komunikace**. Do dalšího stavu S_{12} se obvod dostane okamžitě bez ohledu na vstupy JS.

Stav S_{12}

Ve stavu S_{12} obvod čeká na signál fyzické vrstvy DATAREAD. Další stav je S_{13}. Jinak zůstává obvod ve stavu S_{12}.

Stav S_{13}

Ve stavu S_{13} se na vstup fyzické vrstvy DATA_IN přivede hodnota H'FF' (rámec **Ukončení komunikace**). Do dalšího stavu S_{14} se obvod dostává okamžitě.

Stav S_{14}

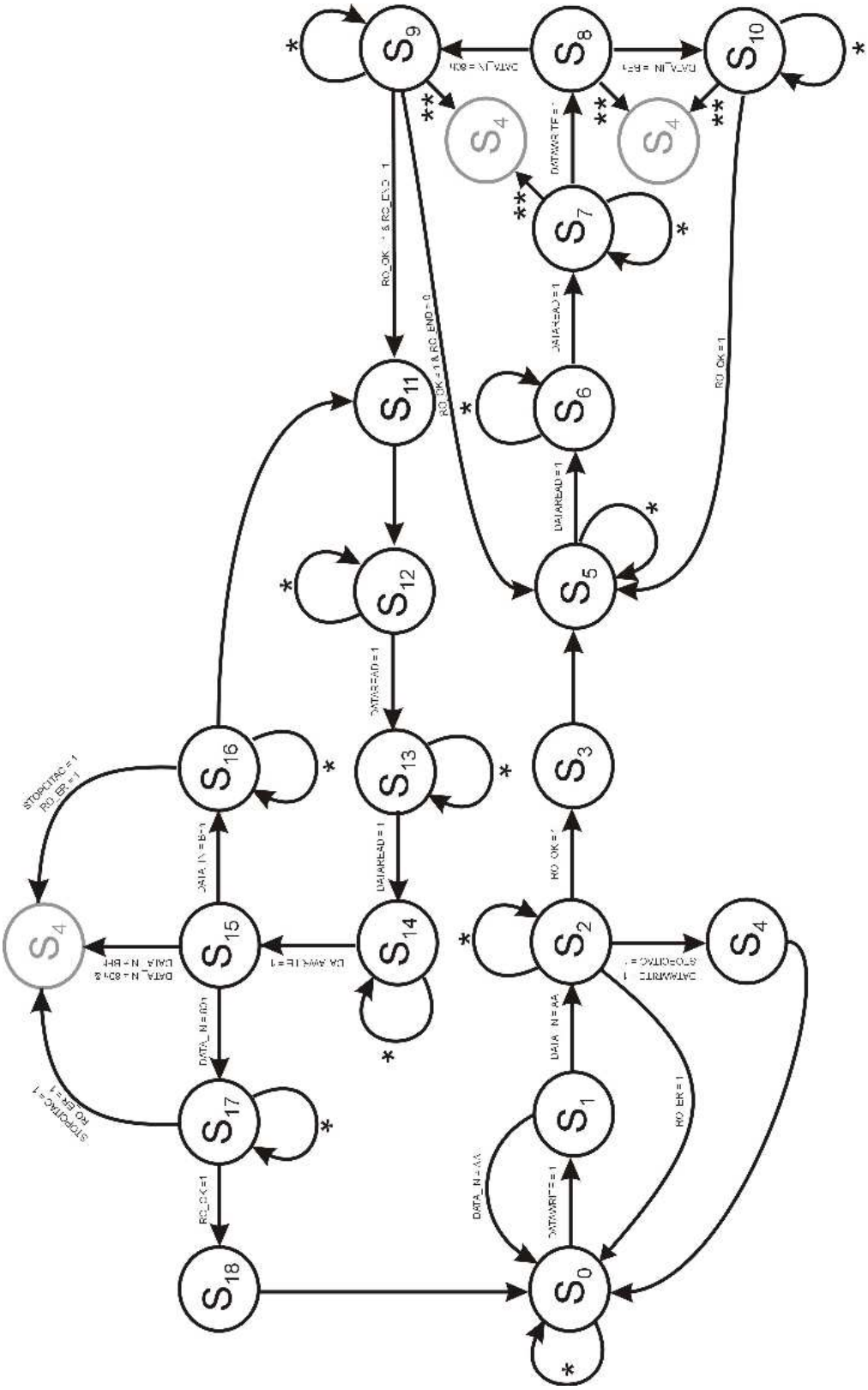
Ve stavu S_{14} obvod čeká na odpověď na rámec **Ukončení komunikace** a na vstup fyzické vrstvy DATA_IN je přivedena adresa JS**.** Do stavu S_{15} se obvod dostane, pokud bude signál DATAWRITE = 1. V případě přetečení čítače se obvod dostane do stavu S_4. Jinak obvod zůstává ve stavu S_{14}.

Stav S_{15}

Do dalšího stavu obvod přechází okamžitě. Pokud je na vstupu DATA_IN hodnota H'80' (**Odpověď na nesprávně přijatý rámec**), je další stav S_{16}. Pokud je na vstupu DATA_IN hodnota H'BF' (**Odpověď na správně přijatý rámec**), je další stav S_{17}.

Stav S_{16}

Ve stavu S_{16} obvod čeká na konec řídicího rámce. Když je výstup z fyzické vrstvy RO_OK = 1, je další stav S_{11}. V případě, že je vstup RO_ER = 1 nebo dojde k přetečení vnitřního čítače obvodu, je další stav S_4.



Obr. 25 – Návrh Moorova stavového automatu pro řídicí obvod mobilní jednotky

* - Hrany označené tímto symbolem naznačují, že řídicí obvod zůstává ve stejném stavu, není-li splněna některá z podmínek pro přechod do dalšího stavu.

** - STOPCITAC = 1

Stav S_{17}

Ve stavu S_{17} obvod čeká (tak jako ve stavu S_{16}) na konec řídicího rámce. Pokud je výstup z fyzické vrstvy RO_OK = 1, je další stav S_{11}. V případě, že je vstup RO_ER = 1 nebo dojde k přetečení vnitřního čítače obvodu, je další stav S_4.

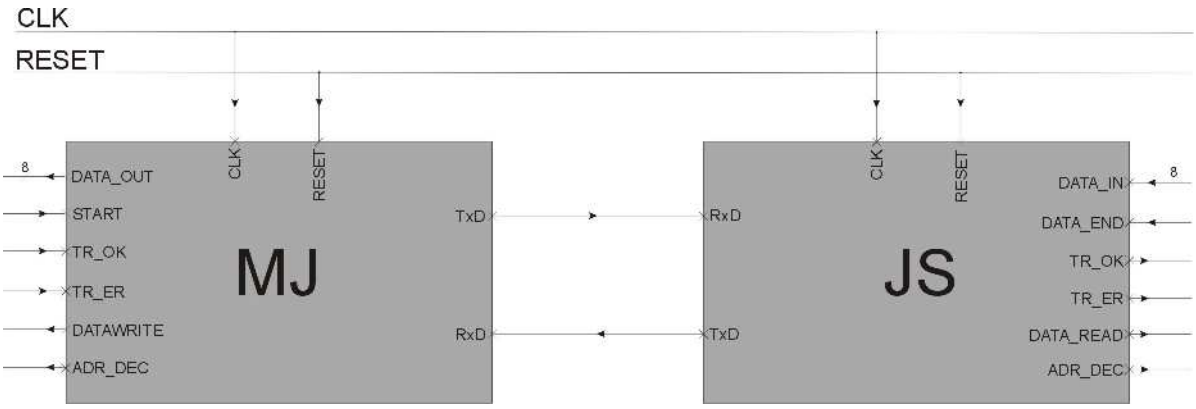
Stav S_{18}

Ve stavu S_{18} je vyšší vrstva informována výstupem jednotky senzoru TR_OK o úspěšném přenosu všech dat z jednotky senzoru. Přenos dat byl uskutečněn v pořádku. Komunikace je úspěšně ukončena. Dalším stav je výchozí stav jednotky senzoru $S_{0,}$ do kterého obvod přechází okamžitě.

8. Komunikace MJ a JS

Při simulaci komunikace MJ s JS byly propojeny oba komunikační obvody podle obrázku 26. Simulace jsou prováděny v prostředí od firmy Altera Max+PlusII na jediném hradlovém poli, ve kterém jsou obsaženy oba komunikační obvody.

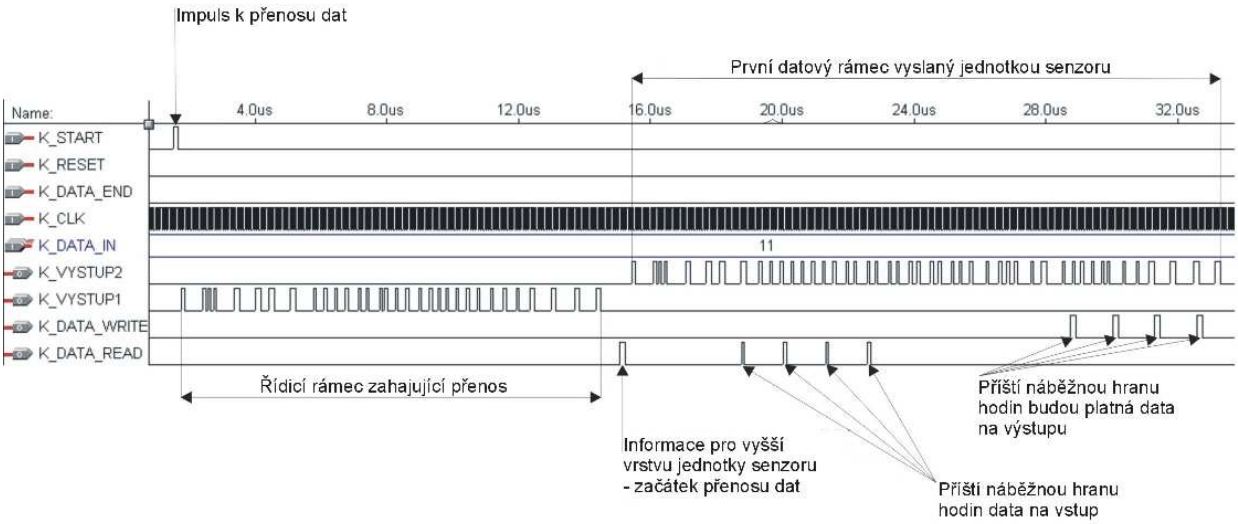
Pro jednodušší průběh simulovaných signálů byla nastavena velikost datových rámců na 5 bajtů (pole Informace má velikost 4 bajty). Sekvence bitů PA (*Preamble Field*) je vysílána pouze 1x (ve standardu IrDA 16x). Tyto parametry přenosu jsou nastavovány ve speciálním souboru obsahující pouze konstanty a lze je jednoduše měnit dle aktuálních požadavků.



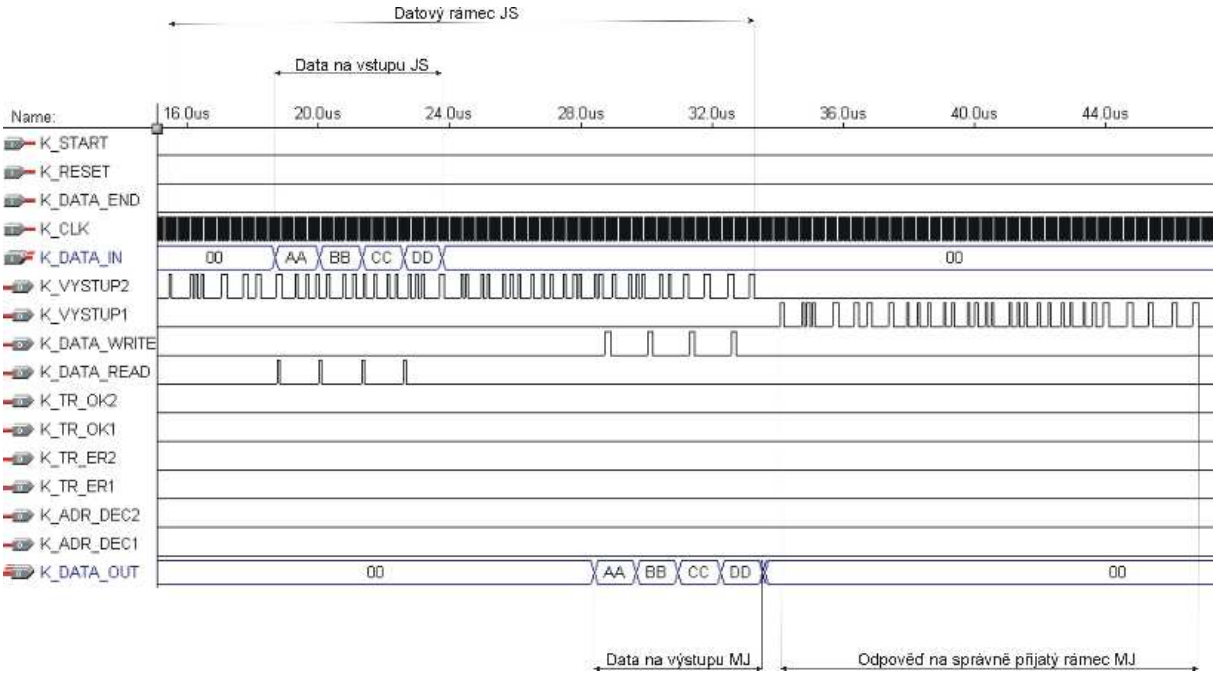
Obr. 26 – Spojení bloků MJ a JS

Začátek komunikace mobilní jednotky s jednotkou senzoru je zobrazen na obr. 27. Po spuštění signálem START mobilní jednotka vyšle rámec **Navázání komunikace**. Jednotka senzoru v této části simulace na navázání komunikace zareaguje vysláním prvního datového rámce přenosu. Výstup označený na obrázku jako K_VYSTUP1 je výstup TxD mobilní jednotky a výstup označený K_VYSTUP2 je výstupem TxD jednotky senzoru.

Průběh vyslání jednoho datového rámce jednotkou senzoru a odpověď mobilní jednotky je na obr. 28. Data na vstup jednotky senzoru jsou přivedena ručně v simulacích s ohledem na požadavky na vyšší vrstvu. V této simulaci je vidět přenos 4 bajtů dat (H‘AA‘, H‘BB‘, H‘CC‘, H’DD‘) ze vstupu jednotky senzoru na výstup mobilní jednotky.

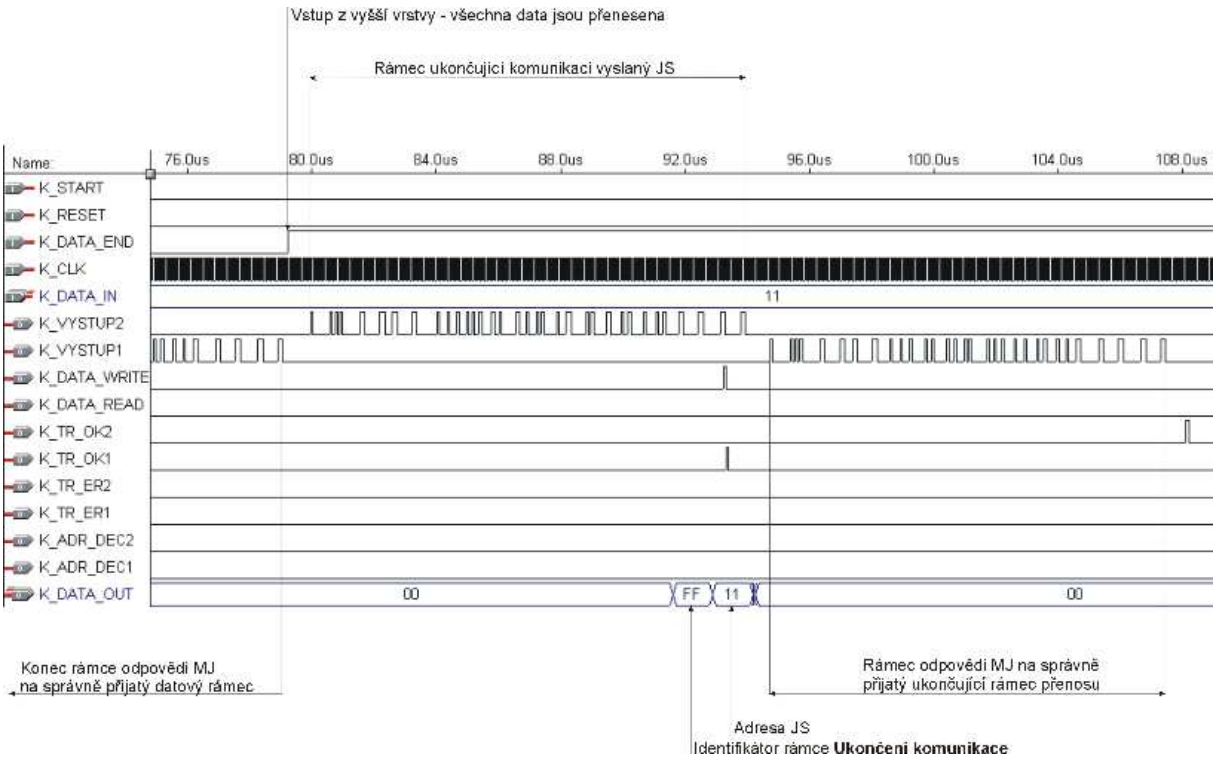


Obr. 27 – Navázání komunikace MJ a JS



Obr. 28 – Přenos dat mezi MJ a JS

Konec komunikace mezi mobilní jednotkou a jednotkou senzoru je na obr. 29. Jediný přenášený bajt v rámci **Ukončení komunikace** je adresa jednotky senzoru. Adresu čte jednotka senzoru tak jako data ze vstupní osmibitové sběrnice. Komunikace je ukončena rámcem MJ, který informuje JS o správném přijmu posledního ukončujícího rámce komunikace. Na výstupech TR_OK1 a TR_OK2 je vidět, jak MJ a JS informují vyšší vrstvu o úspěšném přenosu všech dat z jednotky senzoru.



Obr. 29 – Ukončení komunikace mezi MJ a JS

9. Závěr

Úkolem diplomové práce bylo vytvořit a otestovat řadič pro infračervený přenos dat. Základem pro realizaci přenosu dat po infračerveném rozhraní se stala norma IrDA. Vytvořené elektronické komunikační zařízení obsahuje dvě nejnižší přenosové vrstvy (vrstvu fyzickou a spojovou) a vrstvu aplikační. Fyzická vrstva byla vytvořena jako samostatný blok, který přesně odpovídá požadavkům normy IrDA na fyzickou vrstvu s přenosovou rychlostí 4 Mb/s. Spojová vrstva byla vytvořena s ohledem na aplikaci řadiče z pohledu mobilní jednotky a z pohledu jednotky senzoru. Vrstva aplikační je obsažena ve stejném bloku jako vrstva spojová. Je to dáno rozdělením řešení spojové vrstvy z pohledu mobilní jednotky a jednotky senzoru.

Ověření funkčnosti vytvořeného řadiče bylo provedeno pomocí simulací v prostředí MAX+plus II. Oba bloky mobilní jednotka a jednotka senzoru byly spojeny uměle do jednoho celku. Uvnitř tohoto bloku byly propojeny vstupy s výstupy obou jednotek a dalšími vstupy byl řízen přenos dat tak, jak by tomu bylo ve skutečnosti. Tímto způsobem byly provedeny testy funkčnosti spojové a aplikační vrstvy mobilní jednotky a jednotky senzoru. Výsledky simulací spojové vrstvy jsou uvedeny v kapitole 8.

Při testování funkčnosti fyzické vrstvy bylo nejdříve využito možnosti testování v simulačním prostředí MAX+plus II. V další fázi byla fyzická vrstva testována již přímo nahráním bloku do programovatelného hradlového pole od firmy Altera EPF10K20, které je součástí vývojové desky. Blokové schéma vývojové desky je v příloze 1.

Ověření výstupu vysílače fyzické vrstvy bylo provedeno na logickém analyzátoru HP54620A.

Pro testování přijímače fyzické vrstvy byl vytvořen speciální blok, který vysílá paket v pevném tvaru (dále jen CONSTPAKET). Tento blok byl nahrán do programovatelného hradlového pole od firmy LATTICE Semiconductor. Blok CONSTPAKET nahrazuje vysílač fyzické vrstvy. Důvodem pro vytvoření speciálního bloku CONSTPAKET byla velikost bloku fyzické vrstvy (hradlové pole od firmy LATTICE svou kapacitou nedostačovalo). K dispozici bylo tedy při testování pouze jedno dostatečně veliké hradlové pole. Výstup bloku CONSTPAKET byl přiveden na vstup TxD infračerveného přijímače a vysílače IRMS6400. Dokumentace od firmy Infineon k integrovanému obvodu IRMS6400 je k dispozici [4]. Blok fyzické vrstvy nahraný v hradlovém poli EPF10K20 byl připojen k druhému integrovanému obvodu IRMS6400. Blokové schéma zapojení obvodů je zobrazeno v příloze 2. Datový bajt,

který byl obsažen v přijímaném paketu byl zobrazen na segmentovém displeji vývojové desky.

Výsledkem diplomové práce je řadič pro infračervený přenos dat, který může být jednoduše zařazen do většího bloku a zajišťovat bezchybnou komunikaci po infračerveném rozhraní.

10. Použitá literatura

[1] Angerstein, J. – Schairer, W.: Compatible Data Transmission [online]. 1996. Dostupné z: < http://www.irda.org >.

[2] Hlava, J.: Prostředky automatického řízení II: Analogové a číslicové regulátory, elektrické pohony, průmyslové komunikační systémy. 1. vid. Praha: Vydavatelství ČVUT, 2000. ISBN 80-01-02221-8.

[3] Hlavička, J. et al.: Číslicové systémy odolné proti poruchám. 1. vid. Praha: Vydavatelství ČVUT, 1992. Skripta ČVUT.

[4] IRMS6400 Datasheet [online]. Infineon Technologies Inc. 2000-4-26. Dostupné z: <http://www.vishay.com>.

[5] Megowan, P. – Suvak, D. – Knutson, C.: IrDA Infrared Communications: An Overview [online]. [cit. 2002-3-5]. Dostupné z: < http://www.irda.org >.

[6] Peterka, J. Referenční model ISO/OSI [online]. 1999-1-2. Dostupné z: <http://www.novinky.cz/Index/Drobnohled/1552.html>.

[7] Petrolka, J. et al: Serial Infrared Physical Layer Specification [online]. 2001-5-30. Dostupné z: < http://www.irda.org >.

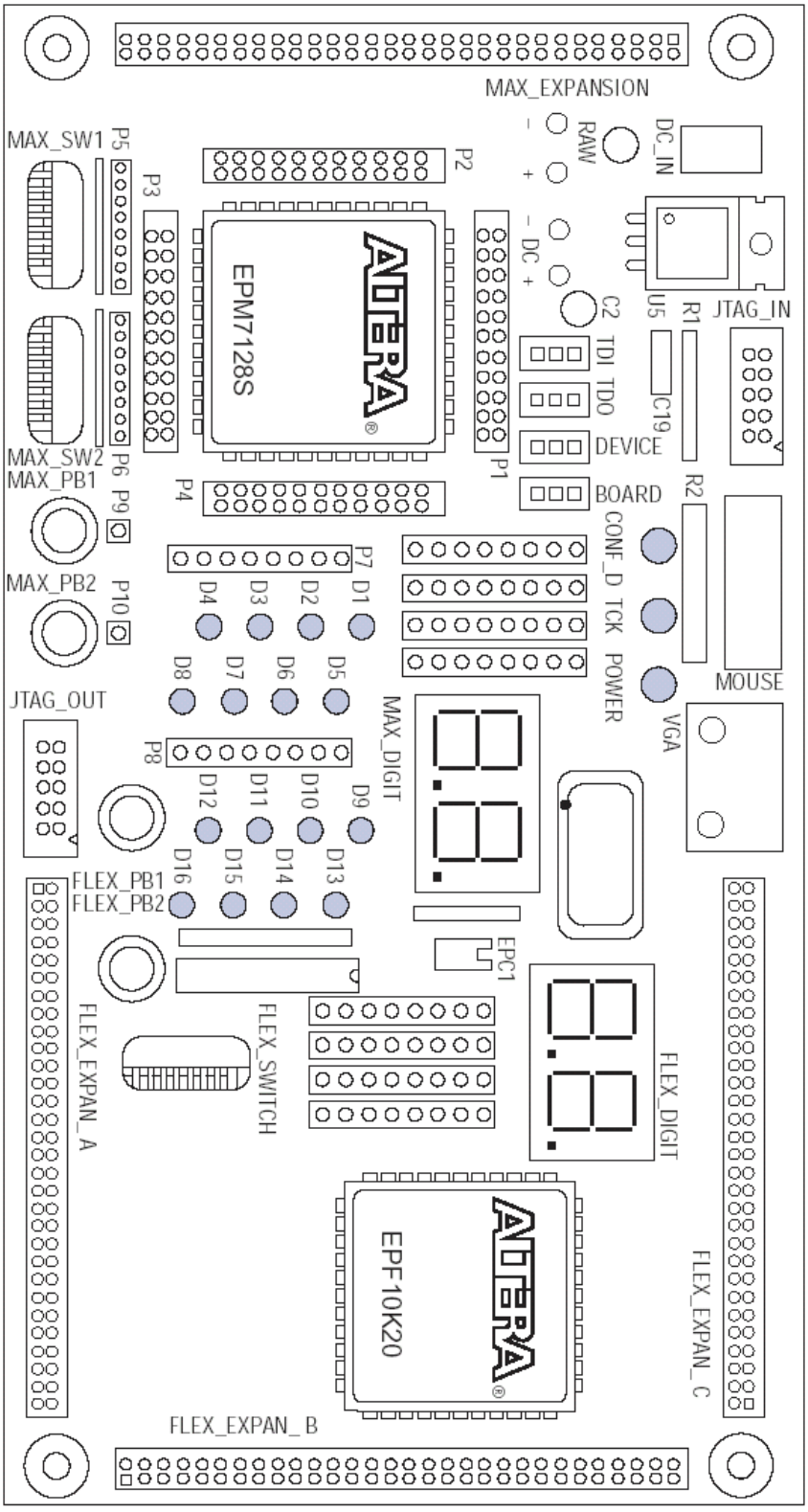
[8] Seaborne, A. et al: Link Management Protocol [online]. 1996-1-23. Dostupné z: < http://www.irda.org >.

[9] Williams, T. et al: Serial Infrared Link Access Protocol [online]. 1996-6-16. Dostupné z: < http://www.irda.org >.

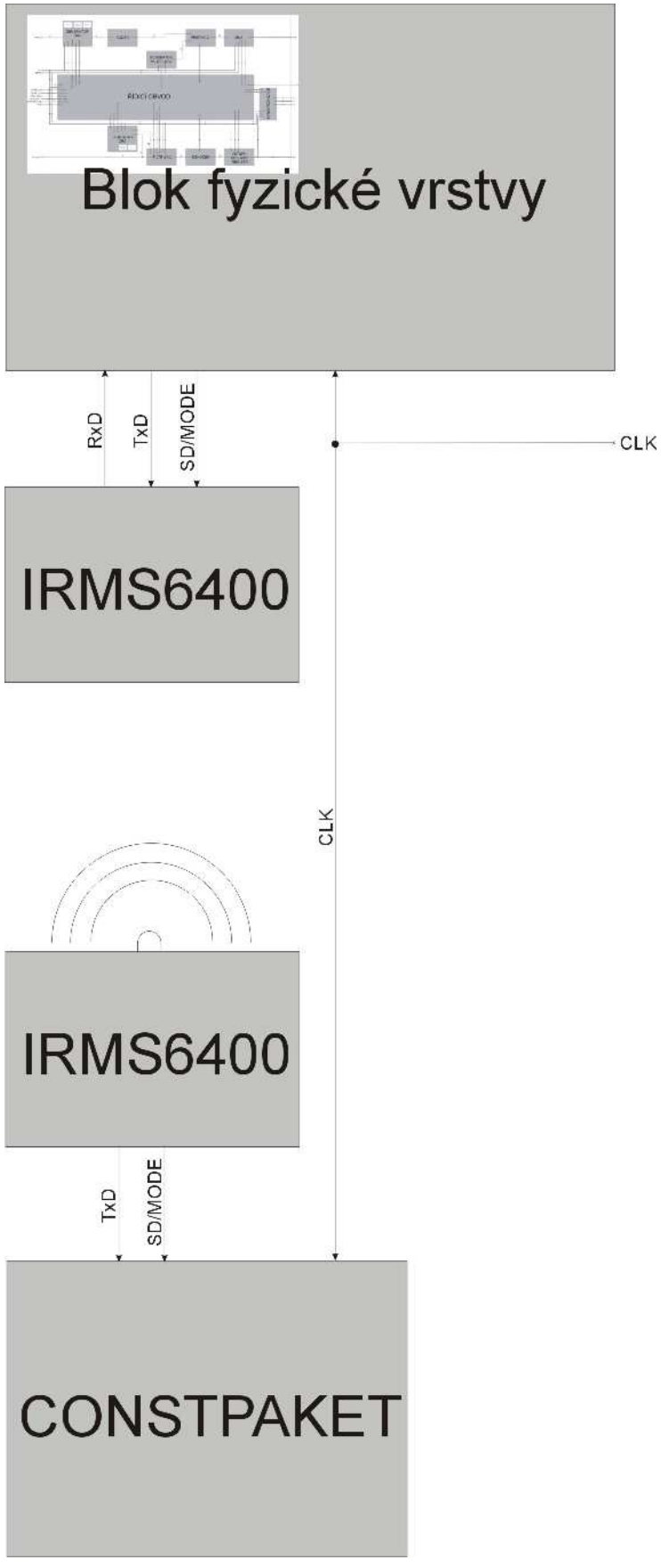
[10] Halama, J. – Hons, D.: Přenos dat infračerveným rozhraním. [Ročníkový projekt]. Liberec 2001. 25 s. Technická univerzita v Liberci. Fakulta mechatroniky a mezioborových inženýrských studií.

Příloha 1: Blokové schéma vývojové desky firmy Altera

Blokové schéma vývojové desky je součástí dokumentace této desky. Dokumentace desky je k dispozici na CD-ROMu v adresáři VÝVOJOVÁ DESKA



Příloha 2: Blokové schéma zapojení při testování přijímače fyzické vrstvy



Příloha 2: CD-ROM

OBSAH CD-ROMu

ADRESÁŘ:

- DiplomovaPrace
  - obsahuje diplomovou práci v elektronické podobě – soubor DP.doc
- IrDA_Normy
  - obsahuje normy IrDA
  - soubory jsou ve formátu pdf
- Irms6400
  - obsahuje dokumentaci k integrovanému obvodu IRMS6400
  - dokumentace je ve formátu pdf
- VHDL
  - tento adresář obsahuje vytvořené VHDL soubory
  - CONSTPAKET
    - obsahuje soubor CONSTPAKET.vhd, který generuje paket v pevném tvaru
    - tento soubor popisuje blok, který byl využíván při testování bloku fyzické vrstvy
  - KOMUNIKACE
    - obsahuje několik soborů, které souží k simulaci komunikace MJ a JS
    - MJ
      - obsahuje soubory, které popisují blok mobilní jednotky
    - JS
      - obsahuje soubory, které popisují blok jednotky senzoru
    - PHY_IrDA
      - obsahuje soubory, které popisují jednotlivé bloky fyzické vrstvy
- VyvojovaDeska
  - obsahuje dokumentaci k vývojové desce od firmy Altera
  - dokumentace je ve formátu pdf